(NASA-CR-143922) ADAPTIVE STATISTICAL PATTERN CLASSIFIERS FOR REMOTELY SENSED DATA Final Report (Tennessee Univ.) 116 p HC $5.25 CSCL 12A

N75-30859

Unclas
G3/65 33049

ELECTRICAL ENGINEERING DEPARTMENT

UNIVERSITY OF TENNESSEE

KNOXVILLE
TN 37916

Final Report Contract NAS8-30878

Project Title

ADAPTIVE STATISTICAL PATTERN CLASSIFIERS
FOR REMOTELY SENSED DATA

Principal Investigator: R.C. Gonzalez
Co-Investigators: M.O. Pace
H.S. Raulston

Technical Report TR-EE/CS-75-10

June, 1975

ABSTRACT

A new technique for the adaptive estimation of non-stationary statistics necessary for Bayesian classification is developed. The basic approach to the adaptive estimation procedure consists of two steps: (1) an optimal stochastic approximation of the parameters of interest and (2) a projection of the parameters in time or position. A divergence criterion is developed to monitor algorithm performance. Comparative results of adaptive and non-adaptive classifier tests are presented for simulated four dimensional spectral scan data.

This work was supported by NASA contract NAS8-30878.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE PAGE

FIGURE PAGE

FIGURE | PAGE

LIST OF SYMBOLS

θ_n = true value of distribution parameter at time (position) n.

Y_n = data sample classified into a particular class at time (position) n.

X_n = "refined" estimate of θ_n made after classification number n provides new data sample.

X^*_n = "projected" estimate of θ_{n+1} made at preceding time (position) n.

$\overline{e_n^2}$ = mean square error $\overline{(X^*_n - \theta_{n+1})^2}$.

γ_{n-1} = weight used in stochastic approximation to specify weighted average of past estimate and present data (chosen to minimize mean-square error, $\overline{e_n^2}$).

$\overline{E_n^2}$ = mean square error $\overline{(X_n - \theta_n)^2}$.

$N(u, \sigma^2)$ denotes normal distribution of mean u and variance σ^2.

CHAPTER I

INTRODUCTION

A Bayes classifier for M pattern classes is essentially a mechanization of M discriminant functions of the patterns $\underline{x}$. These functions are of the form

$$d_i(\underline{x}) = p(\underline{x}/\omega_i)\, p(\omega_i) \qquad (1)$$

$$i = 1, 2, \ldots, M$$

where $p(\underline{x}/\omega_i)$ is the probability density function of the patterns of class ω_i and $p(\omega_i)$ is the *a priori* probability of this class, that is the probability of occurrence of class ω_i. The maximum discriminant function will correspond to the minimum conditional risk. In other words, the Bayes classifier will minimize total expected loss, where loss represents classification error [1].

In order to make a decision on a particular pattern $\underline{x}$, the classifier computes $d_1(\underline{x})$, $d_2(\underline{x})$, ..., $d_M(\underline{x})$, and assigns $\underline{x}$ to class ω_j if $d_j(\underline{x})$ has the largest value. Ties are resolved arbitrarily. Because the Bayes classifier has found such wide acceptance in pattern recognition, this classifier will serve as the basis for an adaptive recognition system capable of adjusting itself to a changing environment.

The structure of a Bayes classifier is determined primarily by the conditional densities $p(\underline{x}/\omega_i)$. Of the various density functions that have been investigated, none has received more attention than the multivariate normal density. Although this attention is due largely to its analytical tractability, the multivariate normal density is also an appropriate model for an important situation: the case where the feature vectors $\underline{x}$ for a given class ω_i represent a single typical or prototype vector $\underline{u}_i$, mildly corrupted by zero mean sampling and measurement noise [2,3].

For M pattern classes the general multivariate normal density functions may be written as

$$p(\underline{x}/\omega_i) = \frac{1}{(2\pi)^{n/2}|C_i|^{1/2}} \exp[-1/2(\underline{x}-\underline{u}_i)' C_i^{-1}(\underline{x}-\underline{u}_i)] \qquad (2)$$

$i = 1, 2, \ldots, M$

n = dimensionality of x

where

$$\underline{u}_i = E[\underline{x}] \qquad (3)$$

and

$$C_i = E[(\underline{x}-\underline{u}_i)(\underline{x}-\underline{u}_i)'] . \qquad (4)$$

For class ω_i the Bayes decision function which minimizes probability of classifier error is found to be $d_i(\underline{x}) = p(\underline{x}/\omega_i)p(\omega_i)$. Due to the exponential form of the normal density function, it is more convenient to work with the natural logarithm of this decision function [1]. The decision function may therefore be written as

$$d_i(\underline{x}) = \text{Ln}\ p(\underline{x}/\omega_i)p(\omega_i) = \text{Ln}\ p(\underline{x}/\omega_i) + \text{Ln}\ p(\omega_i) \tag{5}$$

$$i = 1, 2, \ldots, M.$$

Dropping the term n/2 Ln(2π) because it is common to all M decision functions being compared yields

$$d_i(\underline{x}) = \text{Ln}\ p(\omega_i) - 1/2\text{Ln}|C_i| - 1/2[(\underline{x}-\underline{u}_i)' C_i^{-1}(\underline{x}-\underline{u}_i)] \tag{7}$$

$$i = 1, 2, \ldots, M.$$

An examination of Equation (7) reveals that the changing environment to which the system must adapt is composed of the particular class mean vectors $\underline{u}_i$ and covariance matrices C_i. In the context of a classification, to adapt means to provide the classifier optimal current estimates of parameters necessary for the classification. The parameters may vary with time or position.

In this thesis various stochastic approximation techniques are presented for adaptive estimation. A criterion is also suggested which may be used to detect the divergence of estimates of means.

The ability of these algorithms to accurately estimate the varying mean of a normal density has been tested by computer simulation.

These algorithms have been incorporated into a Bayes classifier to make it adaptive. Comparisons of the various adaptive classifiers, incorporating different estimation algorithms, to the ordinary (non-adaptive) Bayes classifier have been made revealing the desirability of adaptive recognition capability.

A practical application which has been implemented in this work is real-time classification and physical class boundary definition of synthetic multispectral scan data. These boundaries are those between classes in a truth table, and should not be confused with Bayes decision surfaces in pattern space.

The classifier developed here is to serve as a model or prototype; therefore, only the two class recognition problem has been considered. Extension to the more general multiclass case involves no more difficulty than would be involved ith an ordinary Bayes classifier, once the stochastic approximation procedures are understood.

The data used in testing the classifiers was generated on the IBM 360/65 computer system [4]. Algorithm checkout and classifier testing have been performed on the IBM 360/65 and the PDP 11/40 computer systems. Results of classification and subsequent boundary definition have been displayed via the Data Disk video system in conjunction with the PDP 11/40 computer.

CHAPTER II

ESTIMATION ALGORITHMS

A typical sequence of events for classifying and subsequently estimating class statistics at the next time, assuming current estimates have been made, is as follows.

Step 1. The current data sample Y_n is classified into a particular class using current estimates of parameters for all classes.

Step 2. A "refined" estimate of the parameters of the class chosen in Step 1 is computed by stochastic approximation as

$$\theta_n \simeq X_n = X^*_{n-1} + \gamma_{n-1}(Y_n - X^*_{n-1}) \ .$$

This step is omitted for all other classes for lack of data Y_n.

Step 3. A "projected" estimate of θ_{n+1}, X_n^*, may be made by transforming X_n according to the way the algorithm assumes θ is changing with n. If the change is due to time, this step is made for all classes; if the change is due to position (i.e., as when classifying pixels of a multispectral scan frame) within the current class being scanned, this step is performed only for that class chosen in Step 1 above.

Step 4. Increment n by 1 and return to Step 1.

Several notable contributions have been made to the problem of estimating the parameters for a classifier where the class statistics vary with time or space.[1] One such adaptive estimator gave larger weight to more recent samples, as specified by an empirically determined exponential weighting parameter; the consequent "limited memory" made the resultant average more up-to-date [5]. Intuitively the resulting estimates of parameters would be better than an unweighted average. Another adaptive estimation algorithm "projected" the current estimate to the next step by adding an amount of a certain form of anticipated change to the last estimate, and then combining it with the next data sample in a weighted average with weights chosen to minimize the mean square error [6]. This algorithm will subsequently be referred to as the CF algorithm, after the authors. The algorithm developed in this work consists of "refine" and "project" steps [7]. This algorithm differs from the previous one in the sense that the former (1) makes projections suitable for more complex variations with time, and (2) is arranged to operate as part of a Bayes classifier. It will be seen that in both these algorithms the "refine" step of combining previous estimate and new data is in the form of a stochastic approximation formulation shown previously in Step 2 of the typical sequence of events for adaptive classification.

The CF algorithm is essentially a two step algorithm designed to optimally estimate present values of interest rather than to project

---

[1]Time will henceforth denote true time or space (positional index), unless otherwise specified.

an estimate for future use. The two CF steps are defined as follows:

$$(1)\quad X^*_{n-1} = [1 + (n-1)^{-1}]\, X_{n-1} \tag{8}$$

$$(2)\quad X_n = X^*_{n-1} + \gamma_{n-1}(Y_n - X^*_{n-1}) \tag{9}$$

where X^*_{n-1} represents a projected parameter estimate, X_n represents the previous estimate of the parameter, θ_n, and Y_n is the current data sample. γ_{n-1} is a sequence of positive numbers satisfying the conditions of Dvoretzky [8]

$$\lim_{n\to\infty} \gamma_{n-1} = 0, \quad \sum_{n=1}^{\infty} \gamma_{n-1} = \infty, \quad \sum_{n=1}^{\infty} \gamma^2_{n-1} < \infty \tag{10}$$

and chosen to minimize the mean square error of the estimates. Because this algorithm is similar to the form required by an adaptive Bayes classifier, the incorporation of the technique in a classifier is justifiable. In contrast to the empirically derived algorithm discussed previously, this procedure produces optimal estimates.

Examination of equation (8) reveals that this technique assumes the estimated parameter to be time varying in a linear or nearly linear fashion, with zero initial value. The algorithm lacks compensation for an initial non-zero offset or bias of the parameter value.

A modification to the algorithm consisted of subtracting the initial parameter value from the classified sample, applying the CF algorithm to the result, and adding back the initial value to the algorithm estimate. In effect the modification allowed the algorithm to project estimates as if the initial value were zero.

The algorithm developed here produces true distribution parameter estimates for the class of interest at the next classification time. A "refine" step is made, then a "project" step is also made to the next time, because once the data has been classified, the classifier will require an estimate of the future parameter value, not the present. An optimum compromise between the present parameter estimate X^*_{n-1}, made at the previous step n-1, and the present data sample Y_n is made by the stochastic approximation in the "refine" step. The "project" operation then provides the classifier an estimate of the mean for the time when it is actually needed by the classifier. An input, unbiased by variation, is also provided for the next stochastic approximation by the "project" step. Therefore, the "project" operation should remove (in a statistical sense) the estimation bias.

A name considered appropriate for the algorithm is "polynomial fit," hereafter to be referred to as the PF algorithm. The particular algorithm presented was derived to make nonlinear estimates of degree two; however, PF actually represents a class of algorithms derivable for any finite degree. The second degree PF algorithm can be specified as follows.

The refine step (Step 2 of typical classification sequence) is denoted

$$X_n = X^*_{n-1} + \gamma_{n-1} (Y_n - X^*_{n-1}) \approx \theta_n \tag{11}$$

and the project step (Step 3 of typical classification sequence)

$$X^*_n = X_n + \hat{S} \approx \theta_{n+1} \tag{12}$$

where

$$\theta_n \equiv \text{true value at step } n$$

and

$$\hat{S} = \{[i(i+1) - j(j+1)]\, Y_n - [i(i+1)]\, Y_{n-j} + [j(j+1)]\, Y_{n-i}\}/ij(i-j)$$

$$\approx \theta_{n+1} - \theta_n \tag{13}$$

and

$$\gamma_{n-1} = \frac{\overline{e_n^{\,2}} - K_1 \sigma_n^{\,2}}{\overline{e_n^{\,2}} + \sigma_n^{\,2}} \tag{14}$$

and the estimate of mean square error for use in the calculation of γ_n is

$$\overline{e_{n+1}^{\;2}} \equiv \overline{(X_n^{\;*} - \theta_{n+1})^2}$$

$$= \frac{\overline{e_n^{\;2}}\, \sigma_n^{\;2}}{\overline{e_n^{\;2}} + \sigma_n^{\;2}} (K_1+1)^2 + (K_2^{\;2}+K_3^{\;2})\, \sigma_n^{\;2} \tag{15}$$

the required terms for error calculation being

$$K_2 = -\frac{j+1}{i(i-j)} \tag{16}$$

and

$$K_3 = \frac{i+1}{j(i-j)} \tag{17}$$

with K_1 defined as the sum of these two or

$$K_1 \equiv K_2 + K_3 \;. \tag{18}$$

Here the variance of the density function from which samples Y_n are drawn is represented as $\sigma_n^{\;2}$.

Another form of the PF algorithm has been developed using previously projected estimates rather than previous data samples to

fit the polynomial assumed in derivation. This form of second degree algorithm may be specified as follows.

The refine and project steps are specified exactly as in equations (11) and (12) except with

$$\hat{S} = \{[i(i+1) - j(j+1)]\, X_n - [i(i+1)]\, X_{n+j} + [j(j+1)]\, X_{n-i}\}/ij(i-j) \tag{19}$$

and

$$\gamma_{n-1} = \frac{\overline{e_n{}^2}}{\overline{e_n{}^2} + \sigma_n{}^2} \tag{20}$$

and the estimate of mean square error for use in the calculation of γ_n is

$$\overline{e_{n+1}{}^2} = (1+K_1)^2\, \overline{E_n{}^2} + K_2{}^2\, \overline{E_{n-j}{}^2} + K_3{}^2\, \overline{E_{n-i}{}^2} \tag{21}$$

where

$$\overline{E_n{}^2} \equiv \overline{(X_n - \theta_n)^2}$$

$$= (1-\gamma_{n-1}{}^2)^2\, \overline{e_n{}^2} + \gamma_{n-1}{}^2\, \sigma_n{}^2 \tag{22}$$

and the required constants for error calculation being

$$K_2 = \frac{(i+1)}{j(i-j)} \tag{23}$$

and

$$K_3 = \frac{-(j+1)}{i(i-j)} \tag{24}$$

and with K_1 again defined as the sum of these two or

$$K_1 \equiv K_2 + K_3 \, . \tag{25}$$

The "project" operation of equation (12) takes a form suitable for the manner in which the mean is assumed to vary with time while in the CF algorithm, "projection" is accomplished as $X_n^* = (1+1/n)\, X_n$. $\hat{S}$ of equation (12) is an estimate of anticipated change on the next time interval based on the assumption that the true value varies as a second degree function of time, which is in turn estimated by the values of Y_n, Y_{n-i}, and Y_{n-j}, or X_n, X_{n-i}, and X_{n-j}. Equations (13) or (20) give the optimum weight γ_{n-1} to minimize $\overline{e_{n+1}^{\,2}}$ for the two forms of the algorithm. The classifier then uses X_n^* as the best available value for θ_{n+1} for the next classification, at step n+1.

Tests of both forms of the second degree PF algorithm revealed that each made equally reliable projections. A disadvantage of the

second form is that approximately twice as much memory is required in order to accomodate all previous estimates of the necessary error term $\overline{E_i^2}$, i = 1, 2, ..., n.

The ability of the CF and PF algorithms to "track" the varying mean of a Gaussian density has been tested by computer simulation. The data $\{Y_n\}$ were drawn from a unit-variance, one dimensional Gaussian density with mean $9(n-50)^2/2500 + 1$ for n = 1 to 100, and the algorithms produced up-to-date estimates of this mean. Ten statistically independent runs were made for $1 \leq n \leq 100$; the CF algorithm performance is shown in Figure 1, while the second degree PF algorithm performance is shown in Figure 2. For the sake of comparison the performance shown in Figure 3 is that resulting from a least mean square error fit of a second degree curve to the set $\{Y_K\}$, K = 1, 2, ..., 100.

Both the PF and CF algorithms have been applied to the problem of adapting to changing mean vectors and covariance matrices of normal class signatures. In order to adapt to changing covariance matrices, the problem addressed was that of estimating elements of the correlation matrices separately from the elements of the mean vectors and then combining these to form the particular covariance matrices [9]. A problem initially encountered using both algorithms was that of maintaining a positive-semidefinite covariance matrix.

Based on the assumption that covariance terms vary at a slower rate than mean vector components, satisfactory estimates of the covariance matrices of M classes may be obtained by updating the j^{th} class covariance matrix when P samples have been classified as members of that class in the following manner.

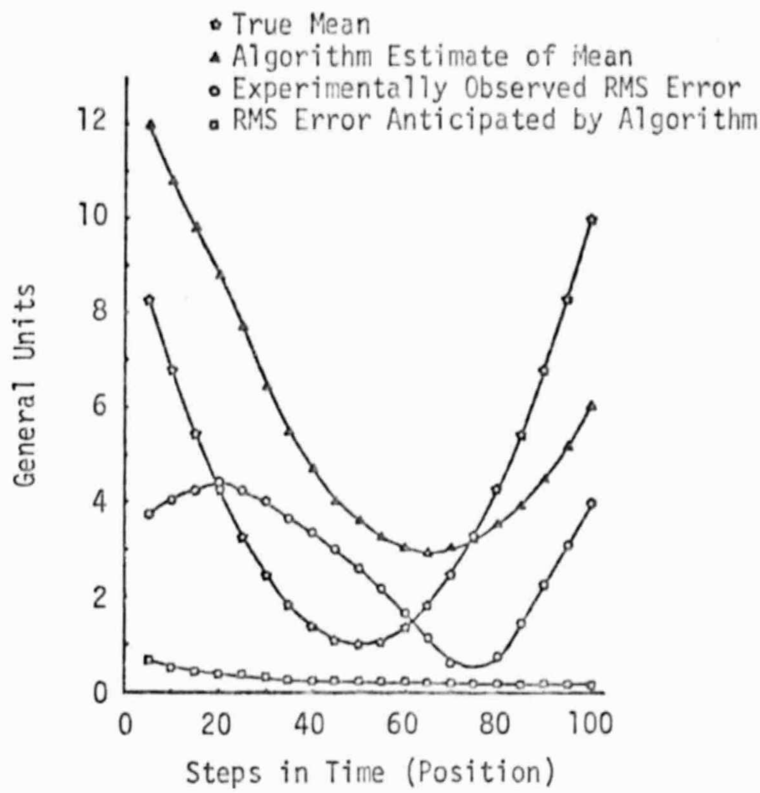

Figure 1. Performance of CF (Chien and Fu) algorithm.

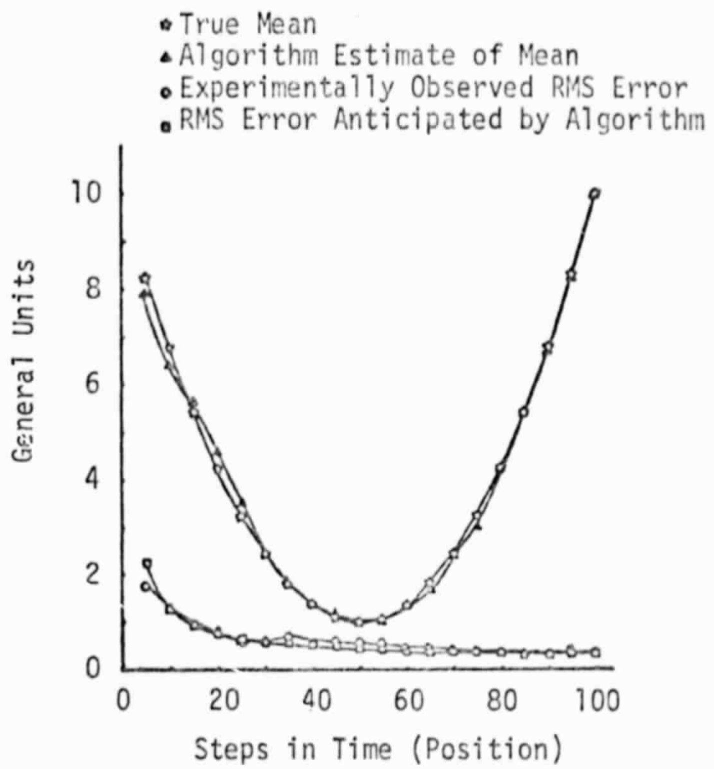

Figure 2. Performance of PF (polynomial fit) algorithm.

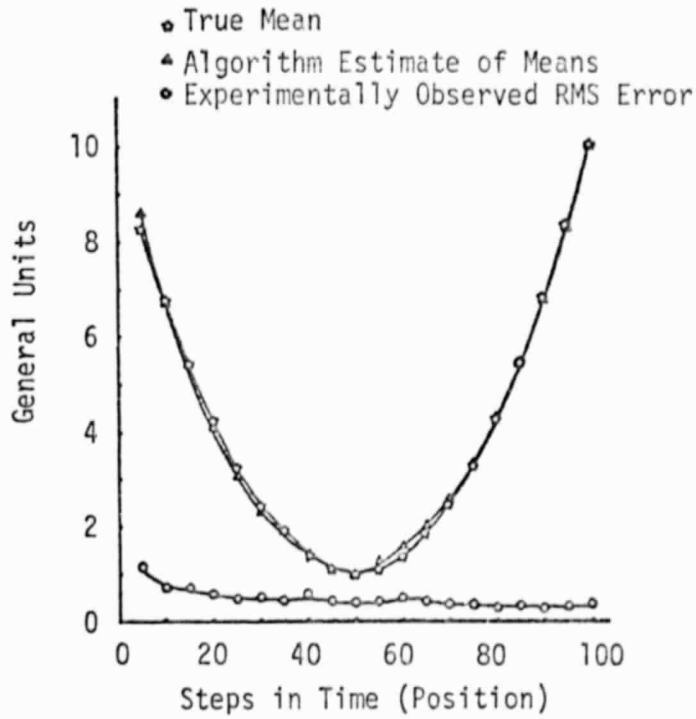

Figure 3. Performance of estimator operating as a least mean square error curve fit.

Step 1 involves specifying the initial covariance matrices for M classes C_i, $i = 1, 2, \ldots, M$, a zero matrix ϕ_i, $i = 1, 2, \ldots, M$ of equal dimensionality to the C matrices, and a counter N_i, $i = 1, 2, \ldots, M$. Each counter should be initialized to zero.

In step 2 specify the number of samples P (where P > dimensionality of pattern vectors) to be used in producing new estimates of the covariance matrix of each class.

At step 3 classify a pattern using the covariance estimate C_i, $i = 1, 2, \ldots, M$ for the classification.

During step 4, if the pattern was classified into class j, update ϕ_j according to the relation

$$\phi_i(N_i+1) = \frac{1}{N_i+1} [N_i \ \phi_i(N_i) + N_i \ m_i(N_i) \ m_i'(N_i)$$

$$+ Y(N_i+1) \ Y'(N_i+1) - \frac{1}{(N_i+1)^2} (N_i \ m_i(N_i)$$

$$+ Y(N_i+1)] + (N_i \ m_i(N_i) + Y(N_i+1)' \tag{26}$$

where m_i represents a mean estimate.

Step 5, increment the counter N_j by one.

Step 6, if N_j is less than P, go to step 3, otherwise, go to step 7.

At step 7, replace C_j by ϕ_j. Reset ϕ_j to the zero matrix and rezero the counter N_j. Go to step 3.

P is chosen greater than the pattern dimensionality in order to insure that the estimate of the covariance will possess an inverse given that the samples are drawn from a normal population [1]. Justification of equation (26) is given in Appendix A.

CHAPTER III

A DIVERGENCE CRITERION

A problem associated with the CF algorithm and also the PF class of algorithms is that their derivations assume the parameters to be estimated vary as some finite degree function. A PF algorithm of very high degree, and hence great flexibility is cumbersome to derive and to run; likewise, computer execution time increases as the degree of algorithm complexity is increased. If the parameter being estimated changes with time in a way more complex than assumed by the algorithm, the predictions of stochastic approximation techniques may diverge. Although the "weak memory" inherent in stochastic approximation will compensate somewhat for this problem, it would be desirable to more strongly limit the memory by restarting the algorithm at the point of divergence, resulting in a piecewise implementation of an estimator. A technique for detecting divergence and restarting the particular stochastic approximation algorithm in the area of divergence is necessary. This restart capability should be provided external to the function of the particular stochastic approximation technique being implemented. In other words, what is needed is a "monitor" for the operation of the algorithm.

Consider the problem of the estimation of the unknown mean of some distribution $Y \sim N(\theta(n), \sigma^2)$, where the mean $\theta(n)$ varies with time. To assume that this function $\theta(n)$ might be approximated by segments would not be unreasonable. The quantity X_n will be considered

the approximation to $\theta(n)$ made by a stochastic approximation algorithm. Associated with each time interval is a random variable Y with variance σ^2. The n^{th} sample value of Y shall be referred to as y_n. If the particular stochastic approximation algorithm accurately estimates $\theta(n)$ based on y_n in some region, it is then possible to define a new, time invariant random variable $Z \sim N(u_z, \sigma^2)$, where the samples z_n are given by

$$z_n = y_n - x_n \ . \tag{27}$$

However, if x_n is an accurate estimate of $\theta(n)$, it is clear that u_z will be zero. A statistical inference built around the notion of a "confidence interval" for a known statistic of the distribution function Z may now be made [10,11]. Let the average value of Z be calculated by the algorithm

$$\overline{z} \equiv \frac{1}{n} \sum_{i=1}^{n} (Y_i - X_i) \ . \tag{28}$$

It can be shown that

$$\Pr[-\frac{3\sigma}{\sqrt{n}} < \overline{z} < \frac{3\sigma}{\sqrt{n}}] \approx 1 \tag{29}$$

(see Appendix B). Therefore, the statistic $\overline{z}$, which is nothing more than the average of the difference of the random sample patterns and

the corresponding estimates of their means, may serve as an indication of divergence.

The restart "monitor" may thus be implemented as follows. If the "confidence interval" condition

$$|\overline{z}| < \frac{3\sigma}{\sqrt{n}} \tag{30}$$

is violated, the algorithm should be restarted at that point (n should be reset to one).

The interval in which $\overline{z}$ must lie is reduced in proportion to $1/\sqrt{n}$. The maximum rate at which a stochastic approximation of a quantity may converge to the true value is in proportion to 1/n, the harmonic sequence, and still satisfy equation (10) of Chapter II [8]. If the γ sequence of equations (9) and (11) approaches the harmonic sequence in the limiting case, it would also be desirable to reduce the confidence interval around $\overline{z}$ in proporation to 1/n. However, were the interval around $\overline{z}$ reduced in proporation to 1/n, the probability of divergence would no longer remain approximately equal to one, nor would it remain constant for each value of n.

The effect of the divergence criterion developed above is to increase the sensitivity to divergence as much as possible while maintaining a constant probability of successfully detecting divergence. In particular this technique has the advantage that it may be used to monitor any estimation algorithm, no matter what degree of complexity was assumed in algorithm derivation.

One point of interest concerns the variance of Z. Since accurate estimation forms the basis for the confidence interval concept, inaccurate estimation will result in the variance associated with Z being larger than σ^2. The resulting divergence criterion may be stricter than anticipated. This problem may be circumvented by making the criterion more lax (for example, by increasing the interval length around $\overline{z}$ to $\pm 4\sigma/\sqrt{n}$).

An alternative method for testing for divergence would be to make use of the estimates of mean square error $\overline{e_n^2}$ made by the algorithms as discussed in Chapter II. If $\sqrt{\overline{e_n^2}}$ could be considered a measure of the error between x_n and the true value $\theta(n)$ and σ a measure of the error between y_n and the true value $\theta(n)$ then x_n and y_n should differ at most by $\sqrt{\overline{e_n^2}} + \sigma$. An algorithm restart, with n reset to one, could be made at the point where

$$K|x_n - y_n| > \sqrt{\overline{e_n^2}} + \sigma$$

with K a constant factor (for example, K = 2).

Another possible variation on this idea might be to use both the original divergence criterion (confidence interval) together with this latter relation in combination.

CHAPTER IV

ADAPTIVE RECOGNITION AND BOUNDARY DEFINITION PROGRAM

An adaptive Bayes classifier is realized by incorporating within the ordinary Bayesian classification program estimation operations which optimally estimate statistics for the next classification time. An application suggested was that the adaptive classifier might be useful in locating or defining spatial boundaries (not to be confused with the Bayes decision surface or boundary) between data classes. A physical example would be the definition of the shoreline between a body of water and a land mass; varying means would then correspond to spectral shifts of scan data caused by transition from deep water to shallow water near the shoreline. As a test, different data sets have been generated, each having two equally likely data classes. These data sets are composed of patterns synthetically produced to simulate a 128×128 pixel frame of four dimensional Gaussian spectral scan data.

Adaptive classification and boundary definition programs have been developed which treat each of the 128 individual horizontal rows as a separate, independent classifier test. These programs utilize the CF and the second degree PF algorithms to adapt to changing class mean vectors. Updated estimates of the covariance matrix for each class are made using the recursive estimation technique discussed in Chapter II.

A general flow chart of program operation is shown in Figure 4. Program initialization is accomplished by specifying an appropriate disk file of input data for classification, specifying a disk file to contain output boundary results for video display, specifying initial estimates of the mean vectors and covariance matrices of the two classes, and inputting a decision variable. The process of classification and subsequent boundary definition then begins.

A 128 pattern row of data is read into memory from the input disk file, each pattern of which is four dimensional. Patterns are classified by a Bayesian classification subroutine. The classifier returns the variable ICLASS as a one or a two to indicate that the pattern has been assigned to class one or class two.

In order to determine whether or not a boundary between the two classes has been crossed in a row test, a stack, whose length is assigned by the specification of the decision variable at initilization, is used. ICLASS associated with the first classified pattern of a row is stored and also pushed onto the stack. The value of ICLASS associated with each successive classified pattern is pushed onto the stack. Only when the stack is full may a decision be made as to whether or not a boundary has been crossed. At that time, and subsequent times, each element of the stack is examined; if more than half of the members of the stack have values equal to that of the ICLASS of the first classified pattern of the row, the boundary definition algorithm decides no boundary has been crossed. If more than half of the members of the stack differ from the ICLASS of the

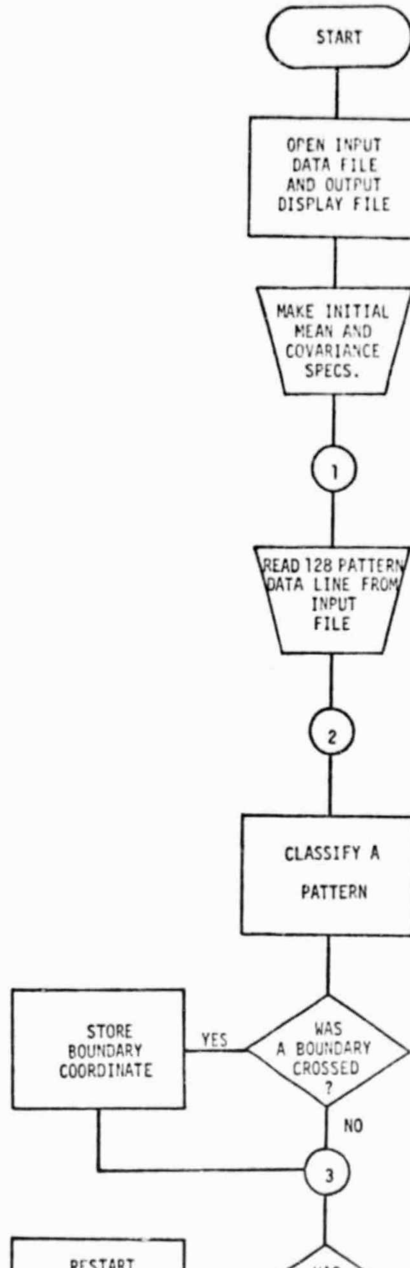

Figure 4. A general flowchart of classification and boundary definition program operation.

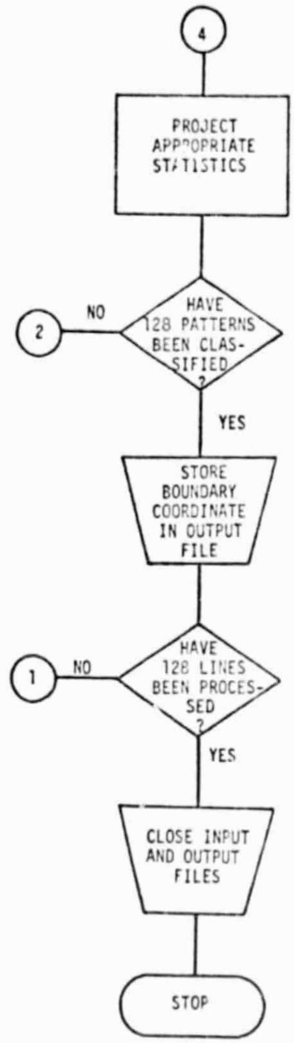

Figure 4. (continued)

first classified pattern of the row, the algorithm decides a boundary has been crossed and the value stored for the ICLASS of the first classified pattern of the row is replaced by the ICLASS of the new class which has been encountered. The appropriate boundary address is stored and the same process continues for the remainder of the row.

After classifying each pattern and performing the boundary test, the divergence criterion of Chapter III may be employed to determine whether or not the estimates of particular mean vector components have diverged. Divergence of a mean vector component requires a restart of the estimation algorithm for that component in the area of divergence.

As each pattern is classified, class statistics for the appropriate class must be projected ahead for the next classification by either the CF or the PF algorithms and the recursive form for the covariance estimation. Upon completion of a 128 pattern row test, boundary information is written into the disk output file, the next row of input data is read, and the process is begun on the unclassified row.

This procedure is repeated until classification and subsequent boundary definition of all 128 rows is accomplished. Upon completion, all input and output disk files are closed and program execution terminates.

Appendices C and D each contain a compiled Fortran IV program listing of two different version of an adaptive Bayes classifier. The numbers at the leftmost side of the listings correspond to the internal

sequence or statement numbers supplied by the Digital Equipment Corporation RT-11 Operating System FORTRAN Compiler. These statement numbers will be used in reference to particular statements.

The first version of the classifier (Appendix C) incorporates the modified CF, algorithm to adaptively estimate class mean vectors, the confidence interval divergence criterion to test for divergence of mean estimates, and the recursive form of covariance estimation. In order to adapt to class mean vectors only and check for their divergence, the statement corresponding to line 117 of the main program should be deleted. To adapt to mean vectors only and neglect the possibility of their divergence, statements corresponding to line numbers 62 through 115 as well as line 117 of the main program should be deleted. To implement the unmodified CF algorithm to adapt mean vectors only, statements corresponding to lines 6, 7, 12, 16, 17, and 22 of SUBROUTINE PROJECT and lines 62 through 115 and also line 117 of the main program should be deleted. An ordinary Bayes classifier (non-adaptive) may be implemented by deletion of lines 62 through 117 of the main program.

The second version of the classifier (Appendix D) incorporates the second degree PF algorithm to adaptively estimate class mean vectors and the recursive form of covariance estimation. In order to adapt to class mean vectors only, the statement corresponding to line 65 should be deleted. To implement an ordinary (non-adaptive) Bayes classifier, the statements corresponding to lines 64 and 65 may be deleted.

CHAPTER V

RESULTS

Five data sets have been synthesized to simulate five 128 × 128 pixel multispectral scan data frames [4]. These data sets are each composed of two classes of four dimensional Gaussian data. A photograph depicting the true spatial boundary between the two classes is shown in Figure 5. The area to the left of this wedge shaped boundary is referred to as class one; similarly, the area to the right of the boundary is class two. The shortest and longest rows of data for each class are 32 and 96 patterns.

Individual rows of data were generated a row at a time from left to right. Data sets one and two were both generated with all four class one mean components varying according to the relation

$$\frac{5}{1024}(N-32)^2 + 5$$

from the left edge of the frame to the boundary (N is simply the position index having an initial value of zero at the left edge of the frame and incremented by one at each position to the right). A plot of this relation versus N is shown in Figure 6. Class two data was generated for the remainder of each row. Class two of data set one was generated having the constant mean vector

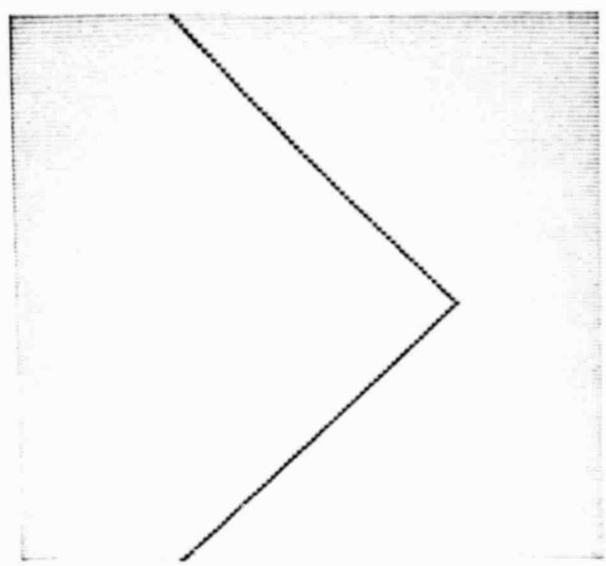

Figure 5. True spatial class boundary.

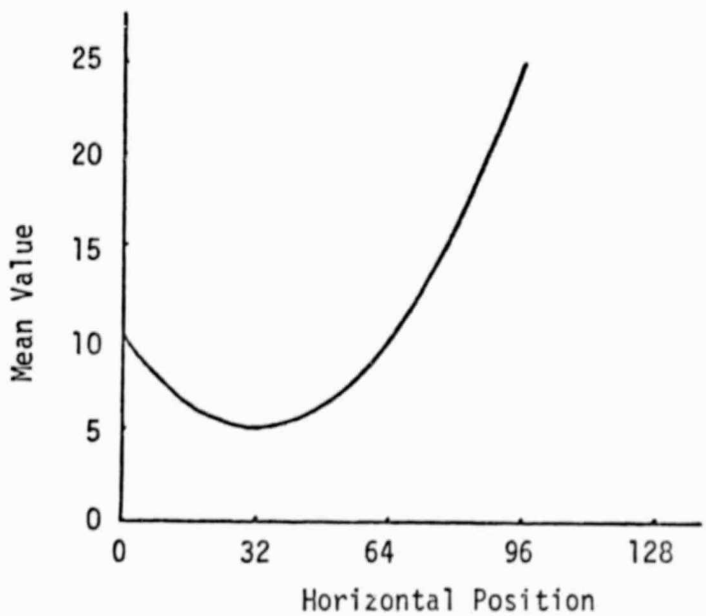

Figure 6. Variation of class one mean with position for data sets 1, 2, 4, and 5.

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

while the mean vector associate with class two of data set two is

$$\begin{bmatrix} 1.5 \\ 1.5 \\ 1.5 \\ 1.5 \end{bmatrix} .$$

The covariance matrices of both classes of data sets one and two are

$$\begin{bmatrix} 1 & .5 & .5 & .5 \\ .5 & 1 & .5 & .5 \\ .5 & .5 & 1 & .5 \\ .5 & .5 & .5 & 1 \end{bmatrix} .$$

Data set three was generated with the four class one mean components varying according to the relation

$$7.5 + 2.5 \cos (.1047N)$$

from the left edge of the frame to the boundary (N again denotes a positional index). A plot of this relation is shown in Figure 7.

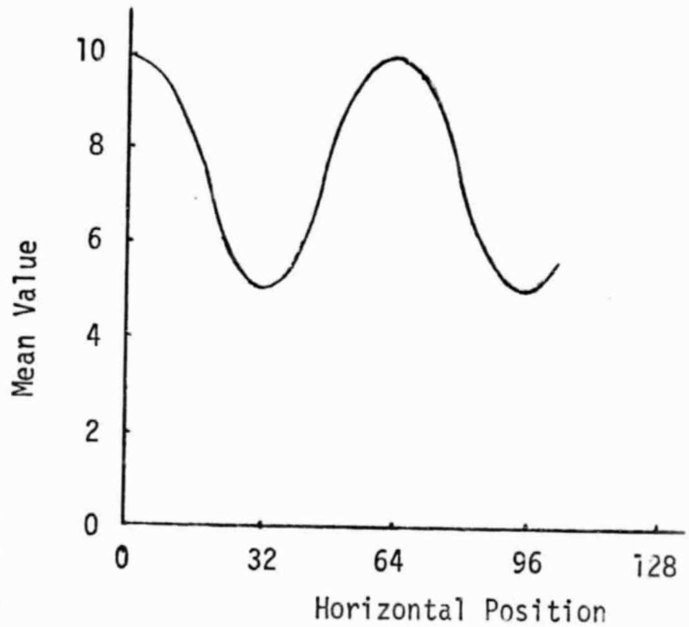

Figure 7. Variation of class one mean with position for data set 3.

Class two data have been generated for the remainder of each row having the constant mean vector

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The covariance of both class one and two of data set three is the same as was specified for data sets one and two.

Data sets four and five were generated having class one and two means specified in exactly the same manner as data set one. In addition, however, each term of the covariance matrix of the class one data was changed in a linear manner according to the equation

$$c_{ij}(N) = c_{ij}(0) + m\,N$$

$$i = 1, \ldots, 4$$
$$j = 1, \ldots, 4$$

where m is simply a slope factor. In other words a linear scalar function of position is added to each term of the initial covariance. For data sets four and five the initial class one covariance was

$$C(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Covariance matrix elements of class one, data set four were varied with a slope m of 0.02 while like elements of data set five changed with a slope of 0.2. Covariance matrices for class two of data sets four and five were both specified as

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Eight different classification and boundary definition programs have been applied to the problem of striking the boundary separating the two classes in each of the five data sets. Each program requires initial estimates of the mean vectors and covariance matrices of the two classes. Because the estimation algorithms predict well, initial mean vector estimates may be made by training over a small area. The effect is to hold sample scatter to a minimum while providing reasonable estimates of the mean.

The first program, referred to as BAYES1, implements an ordinary, non-adaptive Bayes classifier. The initial estimates of class mean vectors and covariance matrices are incorporated throughout classifi-

cation of a complete data set and the resulting data file containing boundary information may be displayed by the DATA-DISK video system.

The second program, BAYES2, employs the CF algorithm in its original form to adaptively estimate mean vectors for a Bayes classifier. Boundary data is subsequently deduced and stored for display.

Program number three, BAYES3, utilizes the modified CF algorithm discussed in Chapter II to produce up-to-date estimates of changing mean vectors for a Bayes classifier.

BAYES4 incorporates not only the modified CF algorithm, but also the confidence interval divergence criterion introduced in Chapter III to adaptively estimate class mean vectors for the classifier.

BAYES5 implements the second degree PF algorithm to adaptively project estimates of class mean vectors for a Bayesian classifier. BAYES6, BAYES7, and BAYES8 take the same form as BAYES3, BAYES4, and BAYES5, respectively, with the exception that BAYES6 through BAYES8 also employ the recursive covariance estimation technique.

Table I provides a cross-reference summary relating Figures 8 through 47 to the particular data sets and programs. Each figure is also individually identified by the program name and data set number used. These figures are photographs of boundaries defined by the various programs for each data set.

A comparison of the results obtained applying the various programs to the different data sets reveals that the ability to adapt to changing mean vectors is essential to successful classification. False boundaries have been generated in each case where the non-

TABLE I

A CROSS-REFERENCE OF FIGURES DEPICTING RESULTS OBTAINED UPON APPLICATION OF THE CLASSIFICATION AND BOUNDARY DEFINITION PROGRAMS TO THE VARIOUS DATA SETS

	DATA SETS				
PROGRAM	1	2	3	4	5
BAYES1	Figure 8	Figure 16	Figure 24	Figure 32	Figure 40
BAYES2	Figure 9	Figure 17	Figure 25	Figure 33	Figure 41
BAYES3	Figure 10	Figure 18	Figure 26	Figure 34	Figure 42
BAYES4	Figure 11	Figure 19	Figure 27	Figure 35	Figure 43
BAYES5	Figure 12	Figure 20	Figure 28	Figure 36	Figure 44
BAYES6	Figure 13	Figure 21	Figure 29	Figure 37	Figure 45
BAYES7	Figure 14	Figure 22	Figure 30	Figure 38	Figure 46
BAYES8	Figure 15	Figure 23	Figure 31	Figure 39	Figure 47

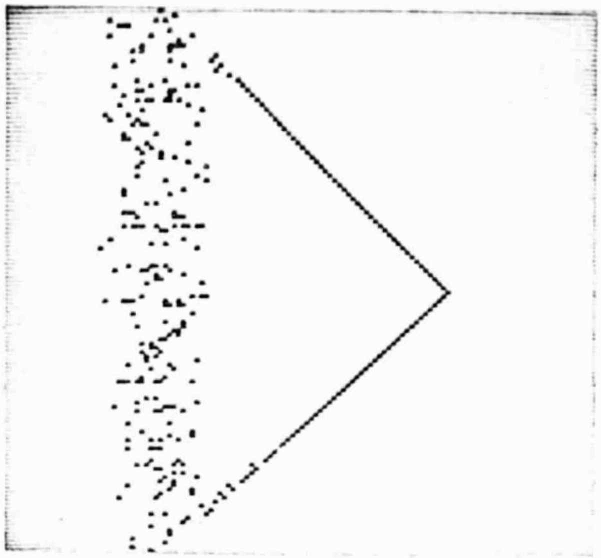

Figure 8. Spatial boundaries resulting from the application of an ordinary Bayes classifier (BAYES1) to data set 1. Note the false boundaries.

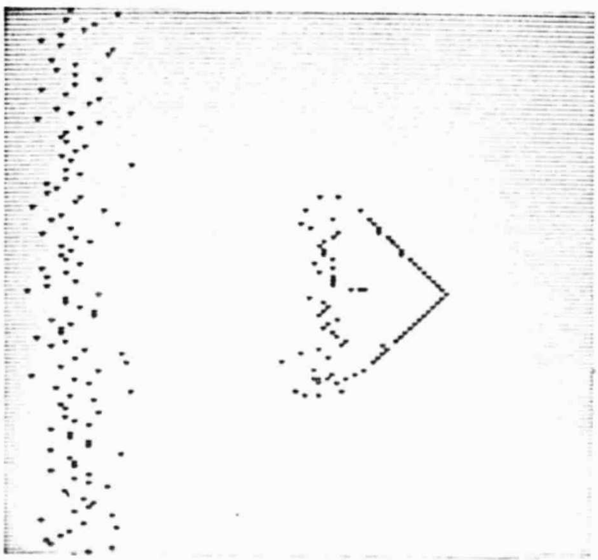

Figure 9. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES2) using the CF algorithm to data set 1. Note the false boundaries.

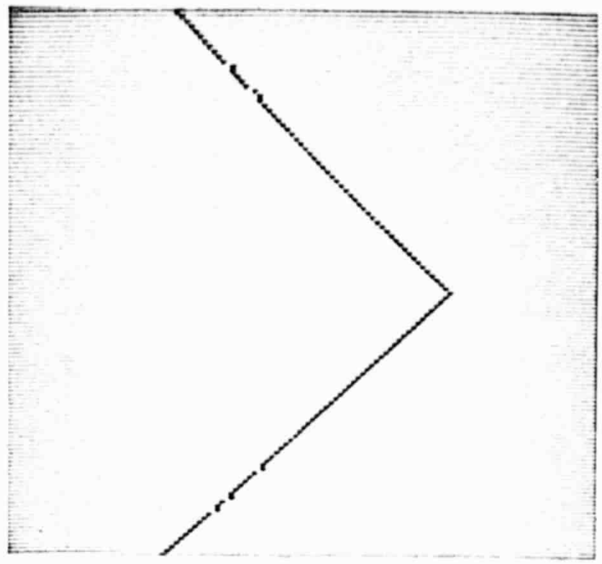

Figure 10. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES3) using the modified CF algorithm to data set 1.

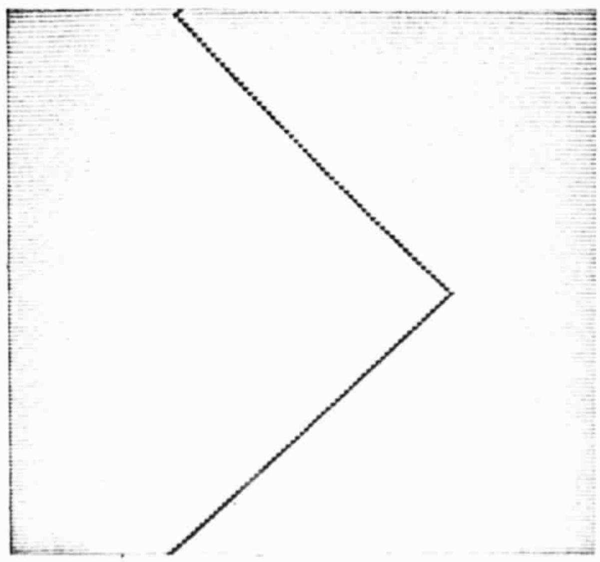

Figure 11. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES4) using the modified CF algorithm and the divergence criterion to data set 1.

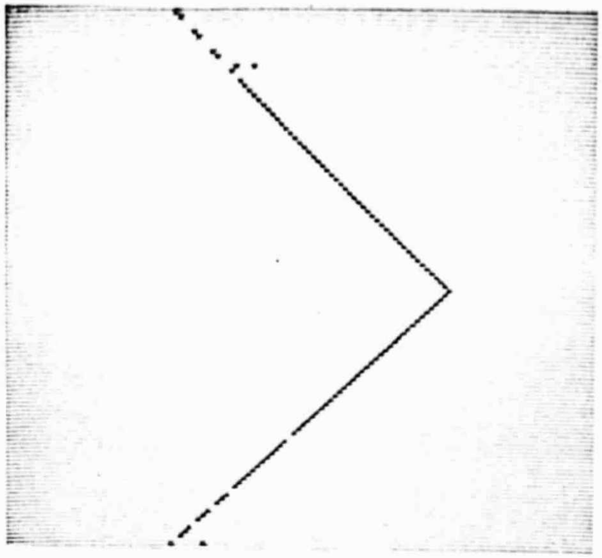

Figure 12. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES5) using the PF algorithm to data set 1.

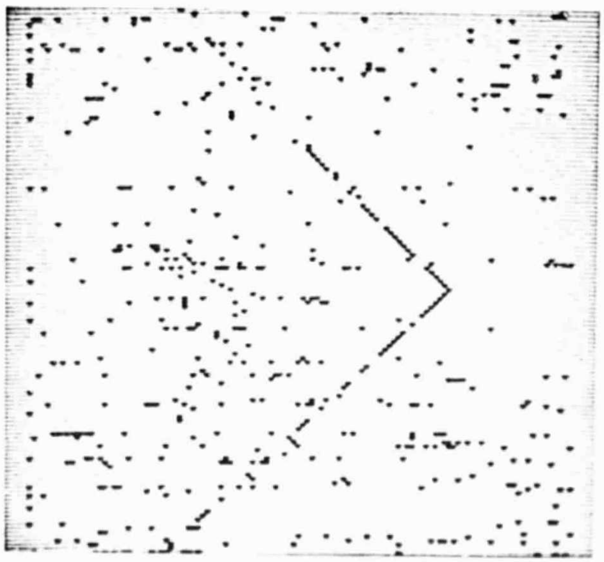

Figure 13. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES6) using the modified CF algorithm and recursive covariance estimation to data set 1.

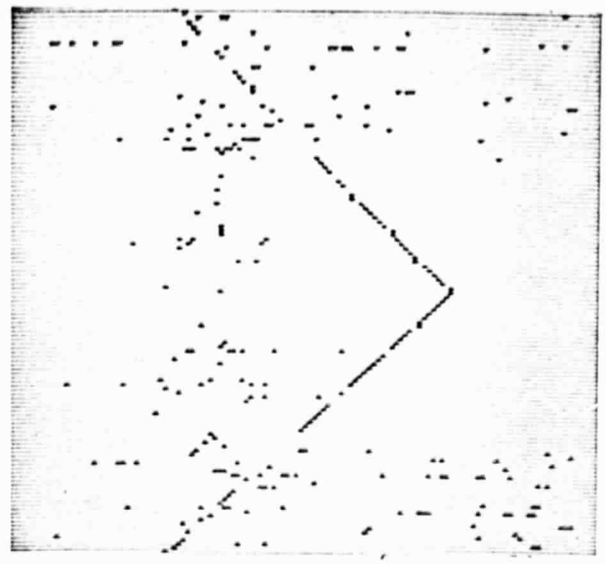

Figure 14. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES7) using the modified CF algorithm, divergence criterion, and recursive covariance estimation to data set 1.

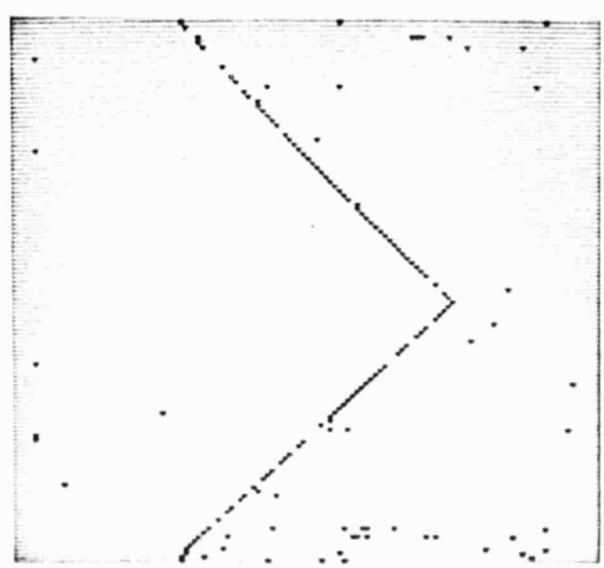

Figure 15. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES8) using the PF algorithm and recursive covariance estimation to data set 1.

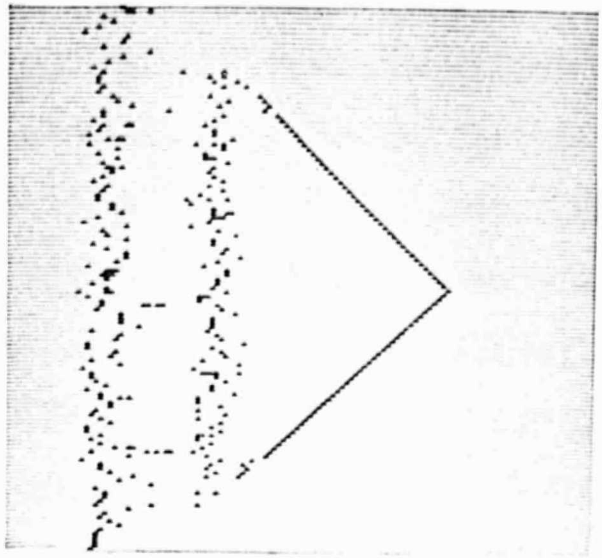

Figure 16. Spatial boundaries resulting from the application of an ordinary Bayes classifier (BAYES1) to data set 2. Note the false boundaries.

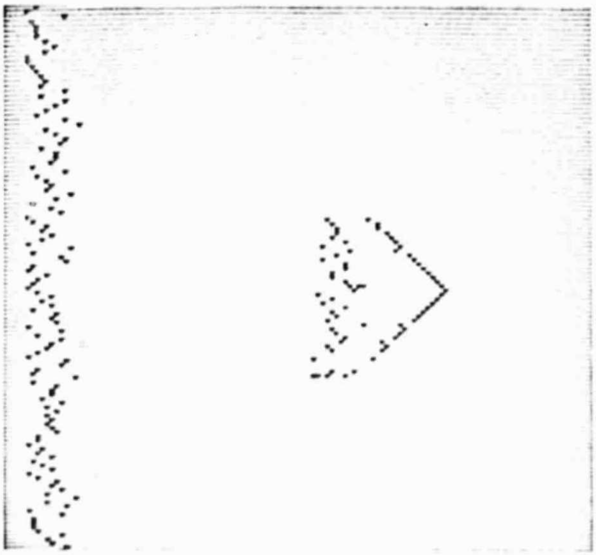

Figure 17. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES2) using the CF algorithm to data set 2. Note the false boundaries.

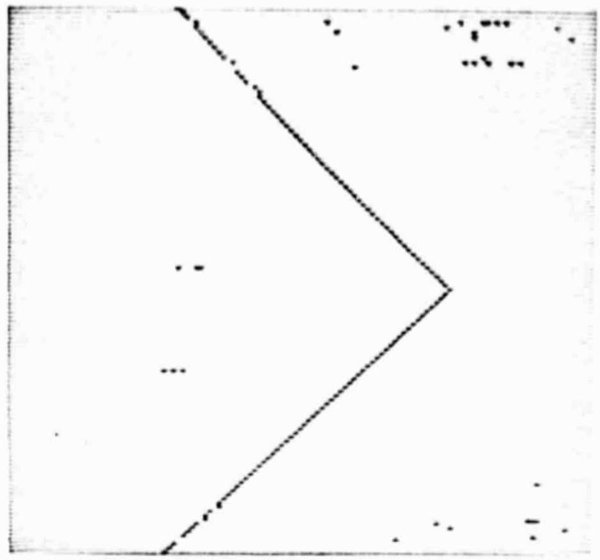

Figure 18. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES3) using the modified CF algorithm to data set 2.

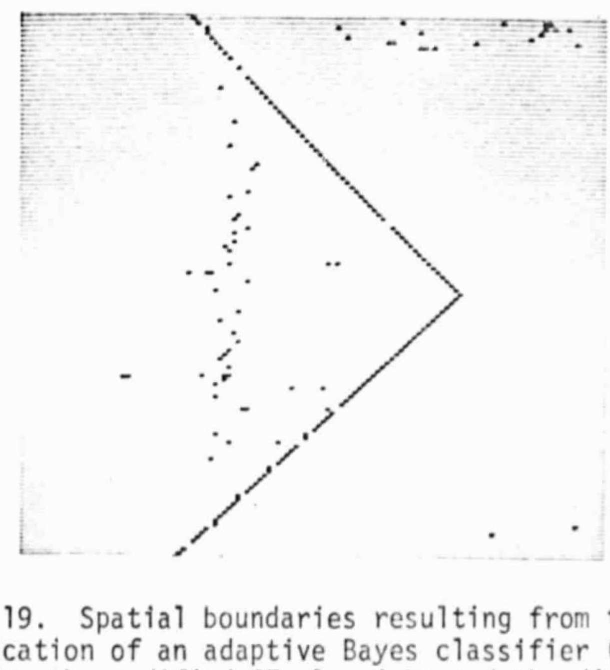

Figure 19. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES4) using the modified CF algorithm and the divergence criterion to data set 2.

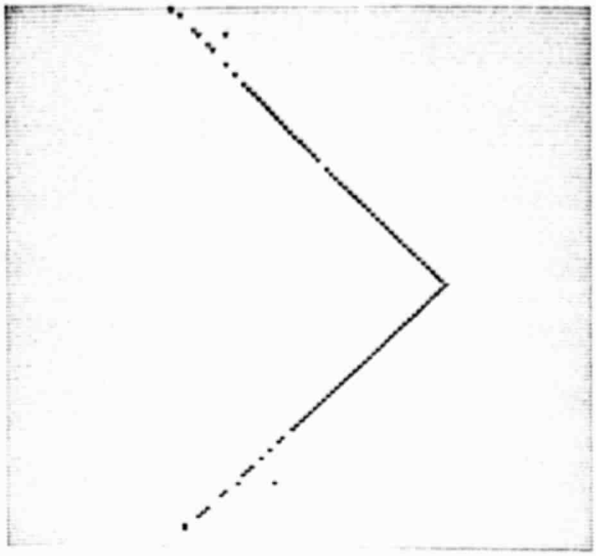

Figure 20. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES5) using the PF algorithm to data set 2.

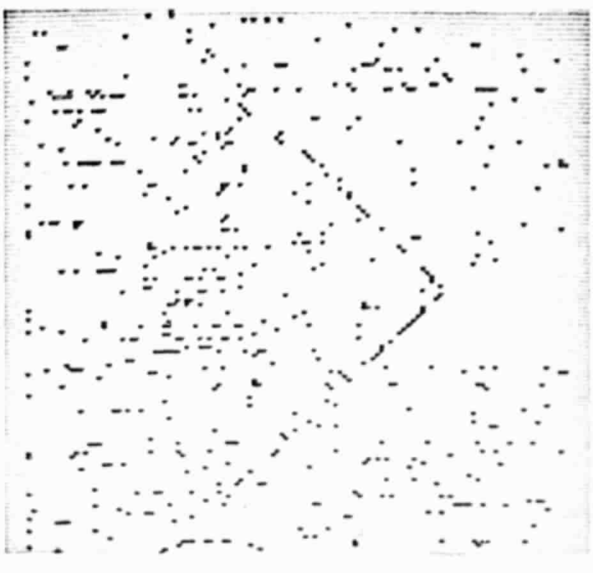

Figure 21. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES6) using the modified CF algorithm and recursive covariance estimation to data set 2.

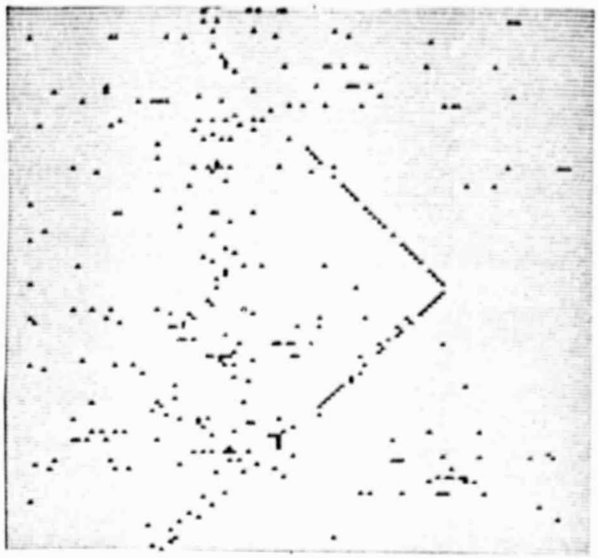

Figure 22. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES7) using the modified CF algorithm, divergence criterion, and recursive covariance estimation to data set 2.

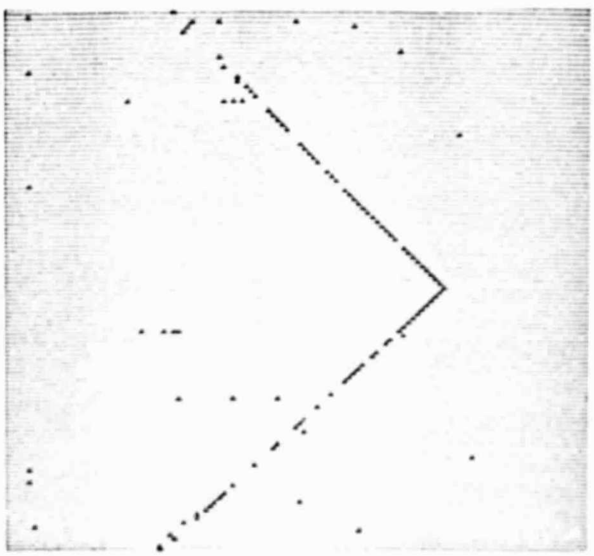

Figure 23. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES8) using the PF algorithm and recursive covariance estimation to data set 2.

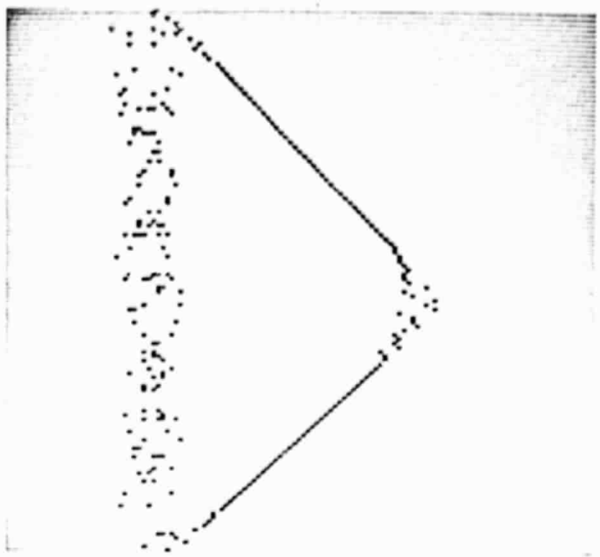

Figure 24. Spatial boundaries resulting from the application of an ordinary Bayes classifier (BAYES1) to data set 3. Note the false boundaries.

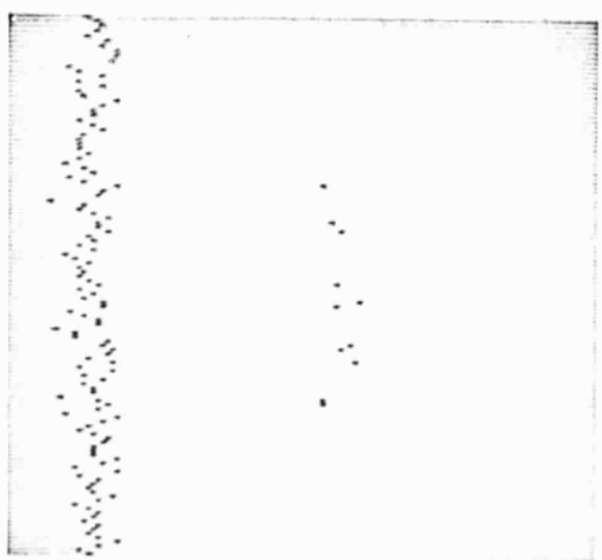

Figure 25. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES2) using the CF algorithm to data set 3. Note the false boundaries.

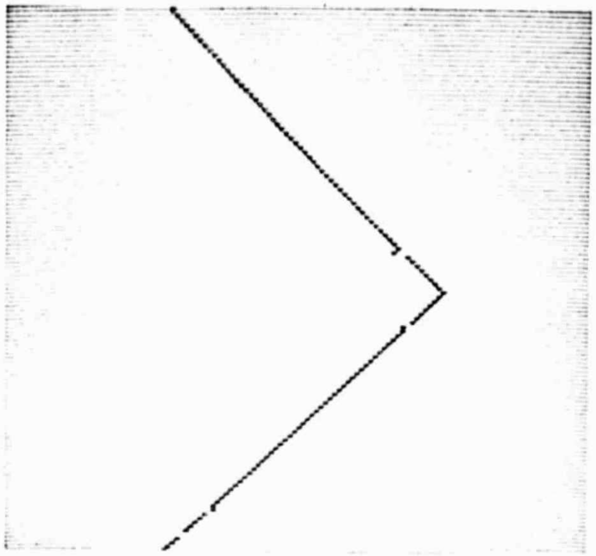

Figure 26. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES3) using the modified CF algorithm to data set 3.

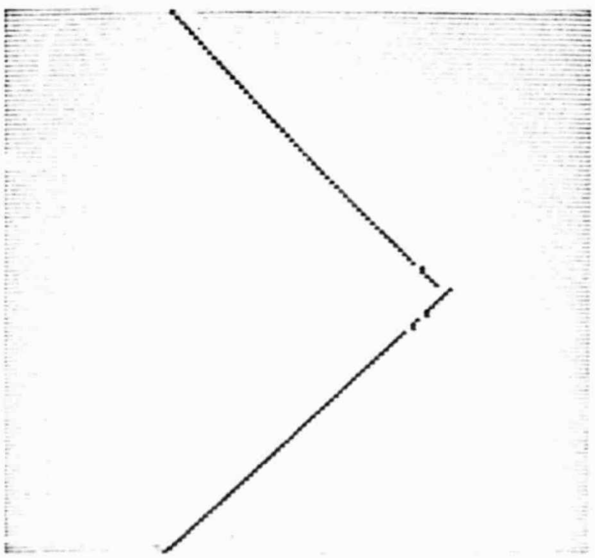

Figure 27. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES4) using the modified CF algorithm and the divergence criterion to data set 3.

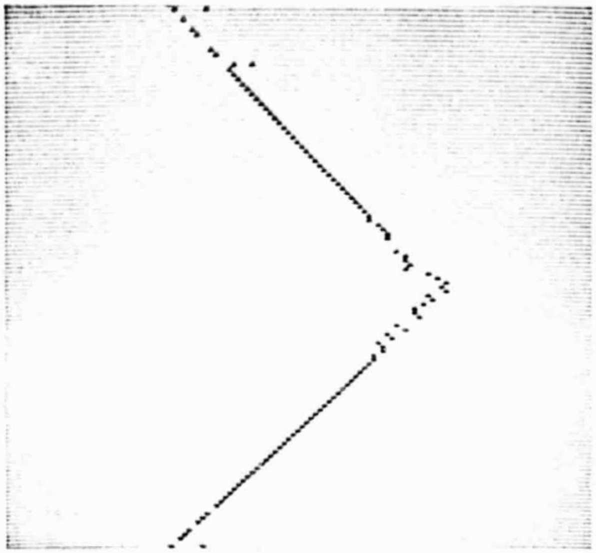

Figure 28. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES5) using the PF algorithm to data set 3.

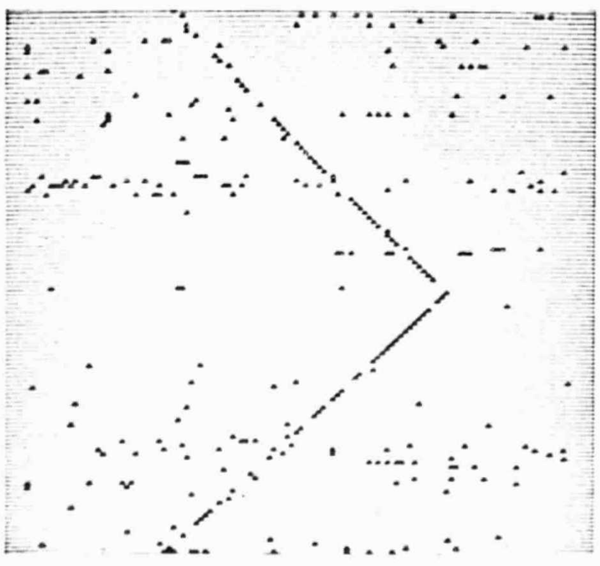

Figure 29. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES6) using the modified CF algorithm and recursive covariance estimation to data set 3.

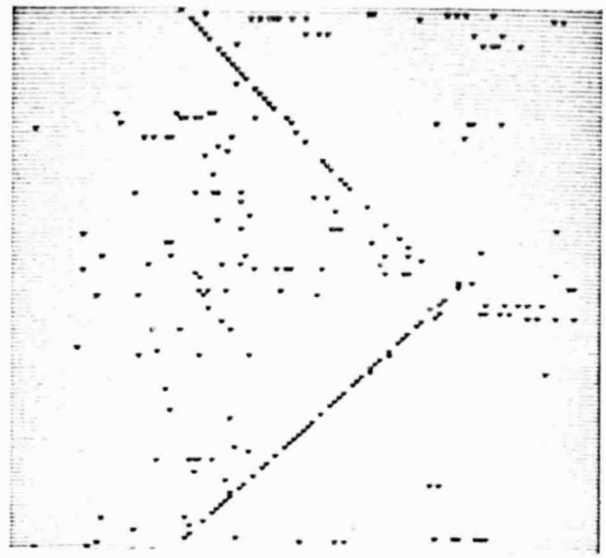

Figure 30. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES7) using the modified CF algorithm, divergence criterion, and recursive covariance estimation to data set 3.

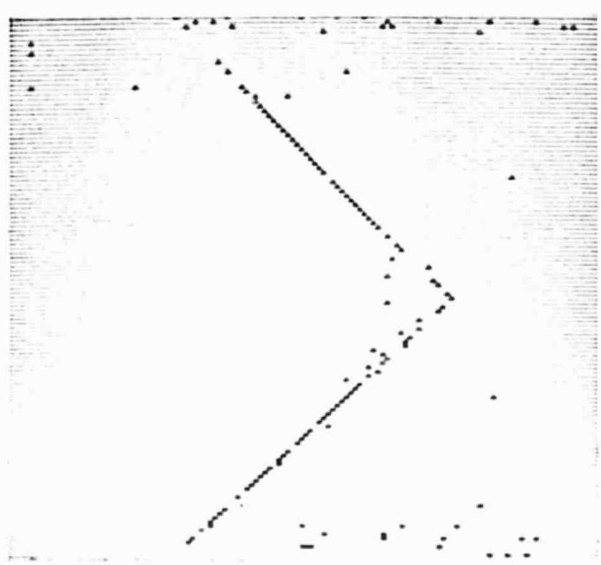

Figure 31. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES8) using the PF algorithm and recursive covariance estimation to data set 3.

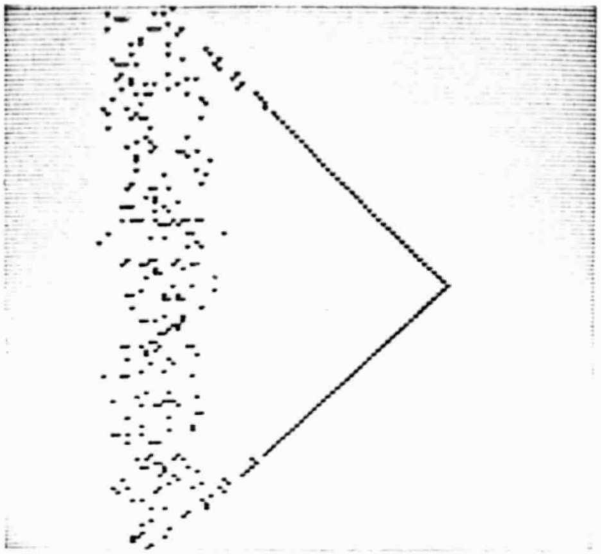

Figure 32. Spatial ! .undaries resulting from the application of an ordinary Bayes classifier (BAYES1) to data set 4. Note the false boundaries.

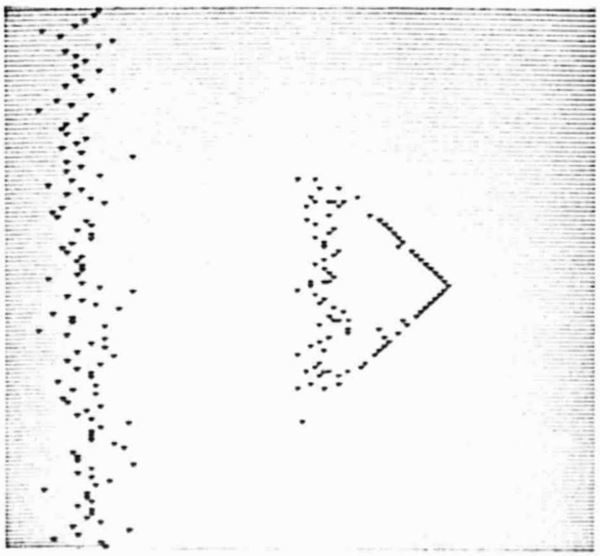

Figure 33. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES2) using the CF algorithm to data set 4. Note the false boundaries.

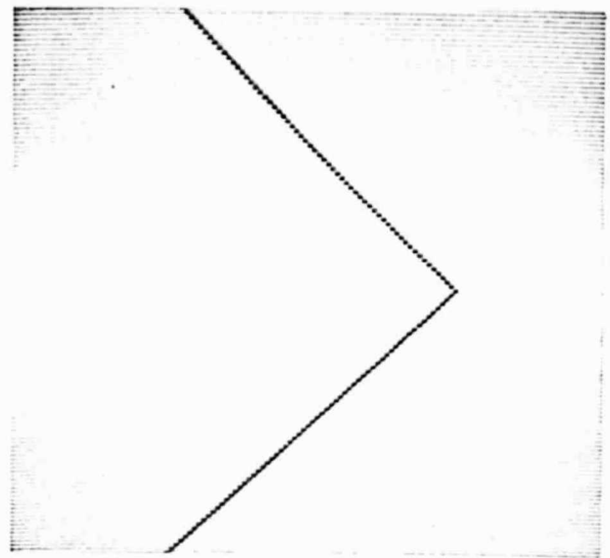

Figure 34. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES3) using the modified CF algorithm to data set 4.

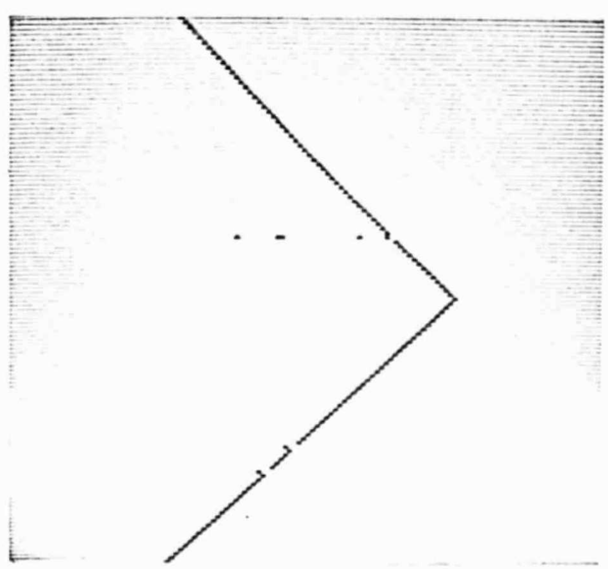

Figure 35. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES4) using the modified CF algorithm and the divergence criterion to data set 4.

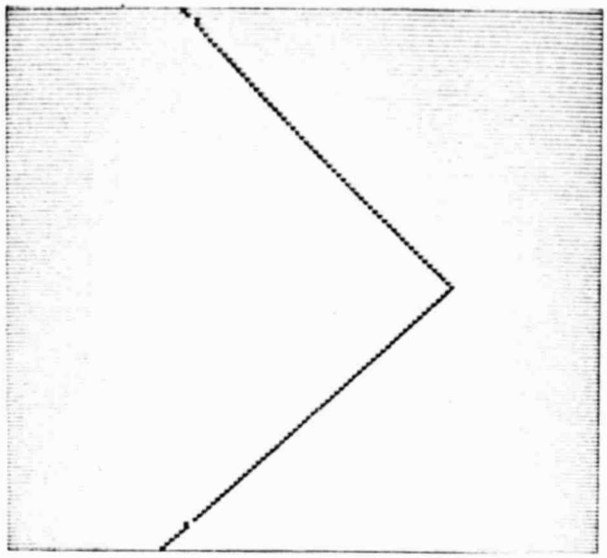

Figure 36. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES5) using the PF algorithm to data set 4.

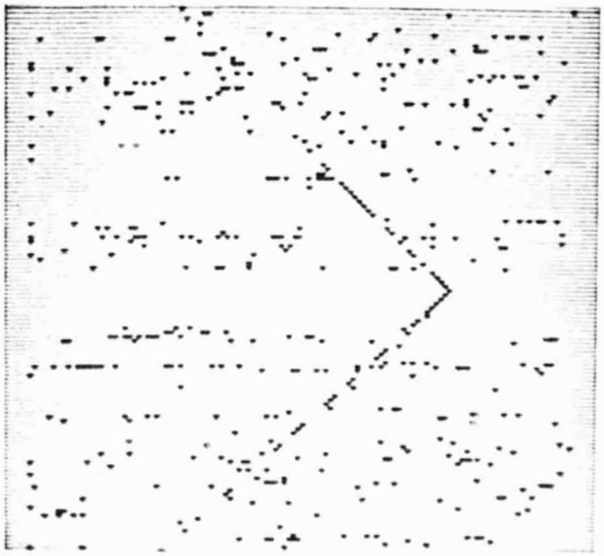

Figure 37. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES6) using the modified CF algorithm and recursive covariance estimation to data set 4.

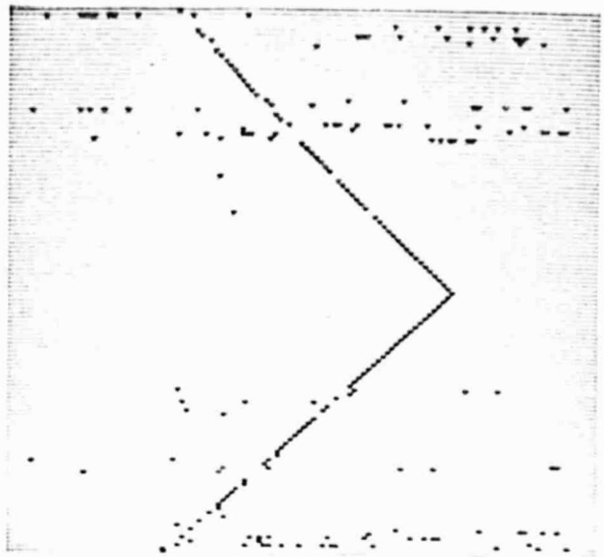

Figure 38. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES7) using the modified CF algorithm, divergence criterion, and recursive covariance estimation to data set 4.

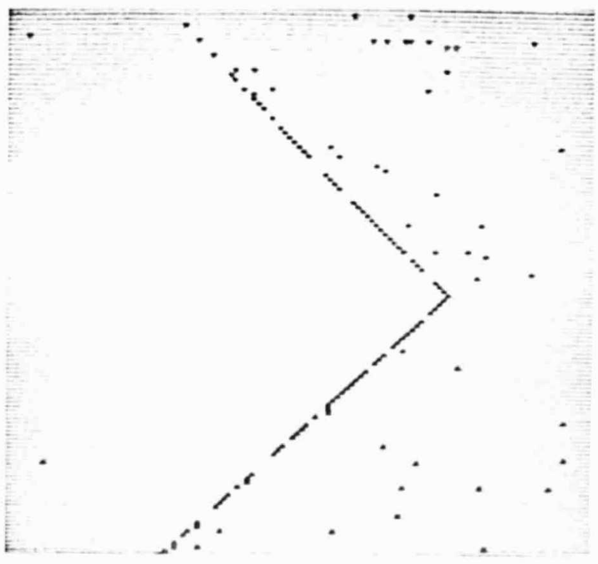

Figure 39. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES8) using the PF algorithm and recursive covariance estimation to data set 4.

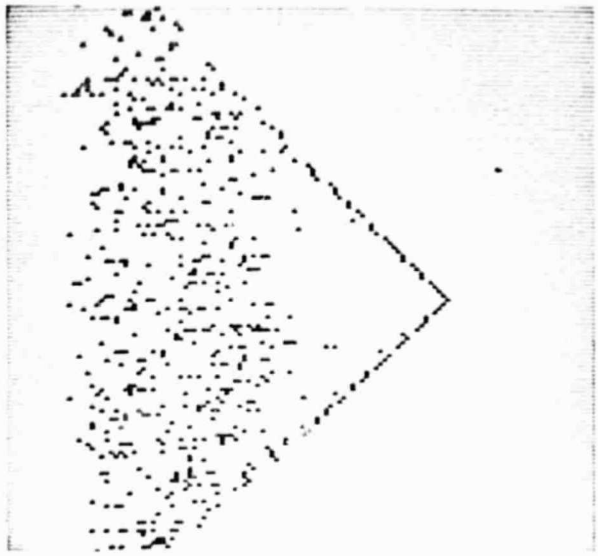

Figure 40. Spatial boundaries resulting from the application of an ordinary Bayes classifier (BAYES1) to data set 5. Note the false boundaries.

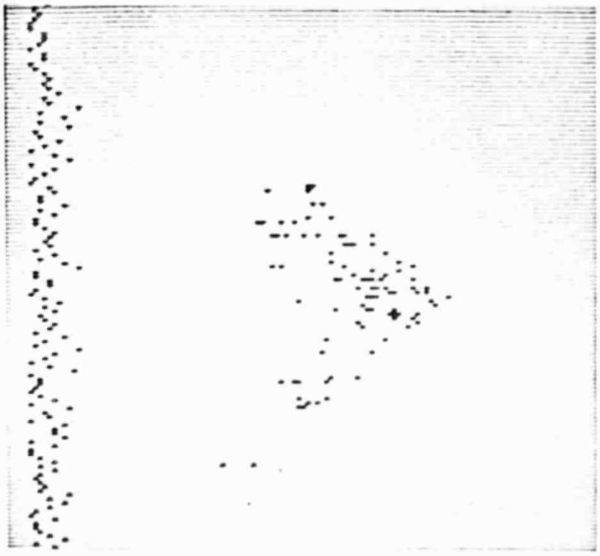

Figure 41. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES2) using the CF algorithm to data set 5. Note the false boundaries.

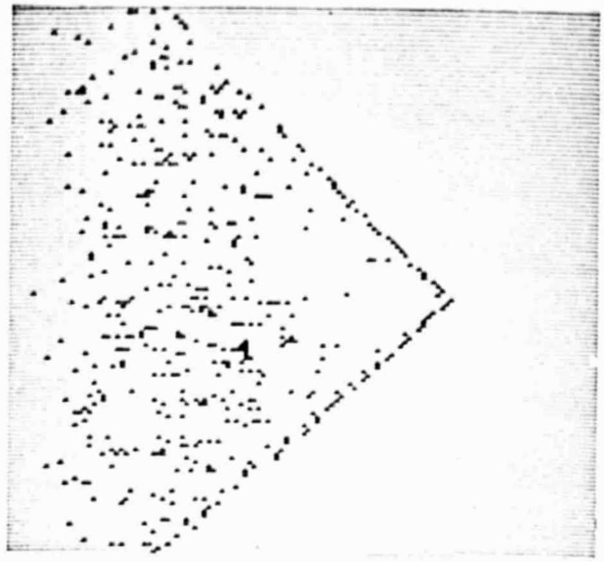

Figure 42. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES3) using the modified CF algorithm to data set 5. Note the false boundaries.

Figure 43. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES4) using the modified CF algorithm and the divergence criterion to data set 5. Note the false boundaries.

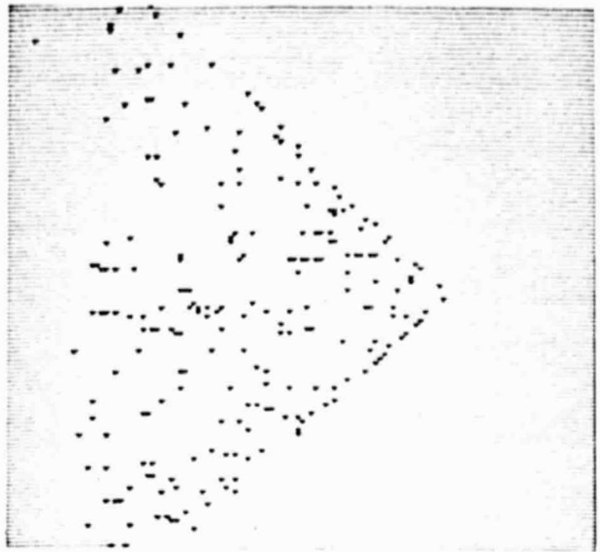

Figure 44. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES5) using the PF algorithm to data set 5. Note the false boundaries.

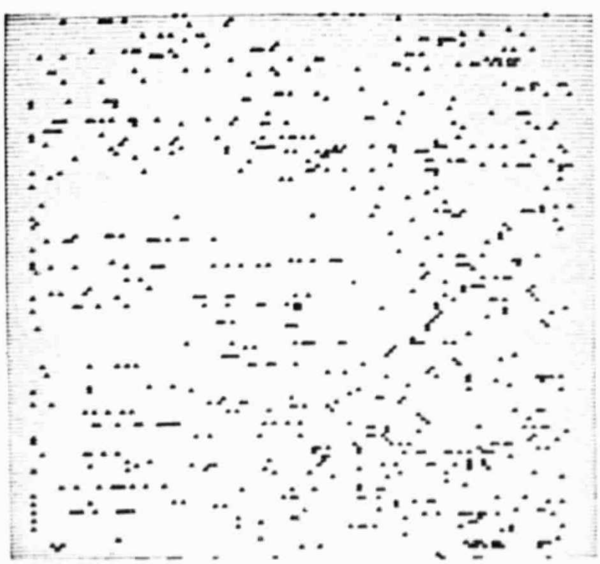

Figure 45. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES6) using the modified CF algorithm and recursive covariance estimation to data set 5. Note the false boundaries.

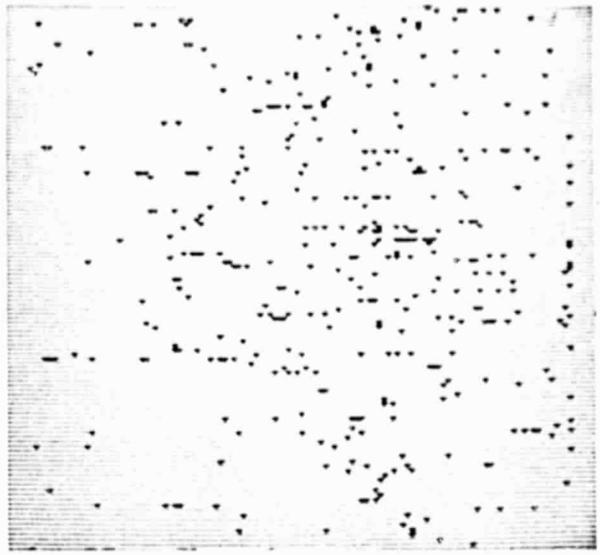

Figure 46. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES7) using the modified CF algorithm, divergence criterion, and recursive covariance estimation to data set 5. Note the false boundaries.

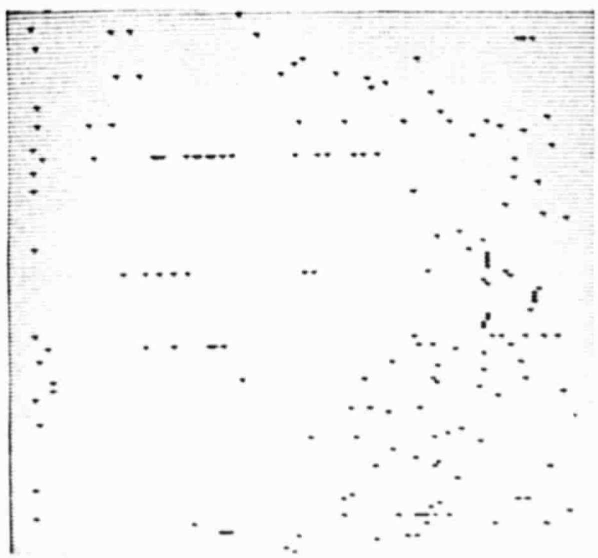

Figure 47. Spatial boundaries resulting from the application of an adaptive Bayes classifier (BAYES8) using the PF algorithm and recursive covariance estimation to data set 5. Note the false boundaries.

adaptive classifier, BAYES1, has been applied to each data set so as to obscure the form of the true boundary. No improvement in the boundary definition results in using the original form of the CF algorithm, implemented in BAYES2, as the adaptive estimator of mean vectors.

Significant improvement in boundary definition performance results have been achieved by BAYES3 which employs a modified CF algorithm to adaptively estimate class means. The modification (see Chapter II) consists of subtracting the initial value of the mean vector of the classified pattern from the pattern and current mean, applying the CF algorithm to the results, and finally adding to the projected mean estimate made by the CF algorithm the initial mean value.

Still more improvement resulted upon application of BAYES4 which incorporates the modified CF algorithm and also the confidence interval divergence criterion of Chapter III. The effect of employing this criterion was to restart the modified CF algorithm (as if n were 1 again) if the condition

$$\left|\frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)\right| < \frac{3\sigma}{\sqrt{n}}$$

was violated. Most of the erroneous boundary points appear at points where restarts were made, due to poor initial tracking when the algorighm is first started with little prior training.

Boundary definition resulting from the application of BAYES5 is very satisfactory, especially in the case of data set two where the degree of overlap is large.

In no case were the results obtained using BAYES6, BAYES7, and BAYES8 of equal quality to the results obtained using BAYES3, BAYES4, and BAYES5. Even in the case of data set four in which a class covariance was changing slowly, the programs which ignored the fact that a covariance matrix could be position dependent proved superior.

In data set five the covariance associates with class one grew at such a rapid rate that class one quickly overlapped class two resulting in an impossible situation for each of the eight techniques.

CHAPTER VI

CONCLUDING OBSERVATIONS

PF type algorithms represent an algorithm class that predicts well; a second degree PF algorithm was used as an example in this thesis but algorithms of this class can be derived (with different $\hat{S}$ and γ formulas) for tracking parameters that vary with time as an n^{th} degree polynomial. In order to achieve the best results, the degree of polynomial assumed in algorithm derivation should be of approximately the same order as anticipated variation. A PF type algorithm can also track variations not of the exact form assumed because of the limited memory characteristic of the "refine" step [7]. Another advantage of the PF class of algorithms is that shifts between algorithms of different complexity can be effected in mid-operation since the output of any PF algorithm (the error estimate and projected parameter estimate) along with the next data sample may serve as the input to any other algorithm of the PF form.

Modifications of the CF algorithm have also been found suitable for tracking varying parameters. Although a PF algorithm may produce better estimates than modified versions of the CF algorithm, especially as the degree of class overlap is increased, the additional memory required for storage of past history may in some cases prohibit use of the PF form and warrant utilization of the modified CF algorithm.

Much work has been done in the area of state estimation in controls engineering. Kalman predictors are capable of producing

statistically optimum state estimates when measurements and inputs are stochastic in nature [12,13]. The Kalman predictor has been found similar in nature to estimation algorithms presented in this thesis. An area for future study lies in investigating the possibility of modifying the Kalman predictor to account for changing covariance.

LIST OF REFERENCES

LIST OF REFERENCES

1. J. T. Tou and R. C. Gonzalez. Pattern Recognition Principles, Addison-Wesley Pub. Co., 1974.

2. R. O. Duda and P. E. Hart. Pattern Classification and Scene Analysis, Wiley-Interscience, 1973.

3. R. B. Crane, W. A. Malila, and W. Richardson. "Suitability of the Normal Density Assumption for Processing Multispectral Scanner Data," IEEE Trans. Geoscience Electronics GE-10, 158-165, 1972.

4. J. K. Bryan and D. L. Tebbe. "Generation of Multivariate Gaussian Date," Dept. of Electrical Engineering, University of Missouri at Columbia, April, 1970.

5. F. J. Kriegler, R. E. Marshall, H. Horwitz, and M. Gordon. "Adaptive Multispectral Recognition of Agricultural Crops," Proc. 8th International Symposium on Remote Sensing of the Environment, 833-849, 1972.

6. Y. T. Chien and K. S. Fu. "Stochastic Learning of Time-Variant Parameters in Random Environment," IEEE Trans. Systems Science and Cybernetics SSC-5, 237-246, July, 1969.

7. H. S. Raulston, M. O. Pace, and R. C. Gonzalez. "Adaptive Bayes Classifiers for Remotely Sensed Data," Symposium on Machine Processing of Remotely Sensed Data, 1A-1-8, June, 1975.

8. D. J. Wilde. Optimum Seeking Methods, Prentice-Hall, 1964.

9. D. G. Keehn. "A Note on Learning for Gaussian Properties," IEEE Trans. Information Theory IT-11, 126-132, January, 1965.

10. R. V. Hogg and A. T. Craig. Introduction to Mathematical Statistics, The Macmillan Co., 1970.

11. M. Dwass. Probability, W. A. Benjamin, Inc., 1970.

12. M. Noton. Modern Control Engineering, Pergamon Press Inc., 1972.

13. A. P. Sage. Optimum Systems Control, Prentice-Hall, Inc., 1968.

APPENDICES

APPENDIX A

COVARIANCE ESTIMATION

Letting C(N) represent the estimate of the covariance for N samples,

$$C(N) = \frac{1}{N} \sum_{j=1}^{N} y_j\, y_j' - m(N)\, m'(N)$$

where the expected value of y has been approximated by the sample average m(N) [1].

$$C(N+1) = \frac{1}{N+1} \sum_{j=1}^{N+1} y_j\, y_j' - m(N+1)\, m'(N+1)$$

$$= \frac{1}{N+1} \left(\sum_{j=1}^{N} y_j\, y_j' + y_{N+1}\, y'_{N+1} \right) - m(N+1)\, m'(N+1)$$

$$= \frac{1}{N+1} \left(NC(N) + N\, m(N)\, m'(N) + y_{N+1}\, y'_{N+1} \right)$$

$$- \frac{1}{(N+1)^2} \left(N\, m(N) + X_{N+1} \right) \left(N\, m(N) + X_{N+1} \right)'$$

This expression provides a convenient method for estimating or updating the covariance matrix, starting with $C(1) = y_1 y_1' - m(1) m'(1)$. Since $m(1) = y_1$, $C(1) = 0$, the zero matrix.

APPENDIX B

CONFIDENCE INTERVAL DERIVATION

Consider the statistic

$$\overline{z} = \frac{1}{n} \sum_{i=1}^{n} z_i .$$

If $\overline{z}$ denotes the mean of a random sample of size n from a distribution $Z \sim N(u_z, \sigma^2)$, then $\overline{z} \sim N(u_z, \sigma^2/n)$ [10,11]. Consider the probability that the interval $(u_z - 3\sigma/\sqrt{n}, u_z + 3\sigma/\sqrt{n})$ includes the point $\overline{z}$. The event $u_z - 3\sigma/\sqrt{n} < \overline{z} < u_z + 3\sigma/\sqrt{n}$ occurs when and only when the event $-3 < \sqrt{n}(u_z - \overline{z})/\sigma < 3$ occurs, thus these two events have the same probability. However, $\sqrt{n}(u_z - \overline{z})/\sigma$ is N(0,1). Accordingly, the probability that the interval $(u_z - 3\sigma/\sqrt{n}, u_z + 3\sigma/\sqrt{n})$ includes the point $\overline{z}$ is equal to

$$\int_{-3}^{3} \frac{1}{2\pi} e^{-z^2/2} \, dz = 0.998 .$$

This probability in no manner depends upon the values of $\overline{z}$, σ^2, or the integer n. Consider next, the length of the interval. The length is seen to be $6\sigma/\sqrt{n}$. Note that this length is unknown until both σ^2 and

n are known; note also that for σ^2 assigned, this length may be made as short as desired by taking n sufficiently large.

For the particular interval considered above, 0.998 was found to be the probability that the interval $(u_z - 3\sigma/\sqrt{n},\ u_z + 3\sigma/\sqrt{n})$ contains the statistic $\overline{z}$. That is,

$$\Pr[u_z - 3\sigma/\sqrt{n} < \overline{z} < u_z + 3\sigma/\sqrt{n}\,] = 0.998 \ .$$

Since σ is known, each of the variables $u_z - 3\sigma/\sqrt{n}$ and $u_z + 3\sigma/\sqrt{n}$ is a known quantity if u_z is known.

APPENDIX C

COMPILED FORTRAN IV PROGRAM LISTING OF ADAPTIVE BAYES CLASSIFIER INCORPORATING MODIFIED CF ALGORITHM AND CONFIDENCE INTERVAL DIVERGENCE CRITERION

```
C
C
C     THIS PROGRAM READS A 128 X 128 DATA POINT ARRAY FROM A DISK
C     FILE, EACH POINT OF WHICH MAY BE CONSIDERED A FOUR DIMENSIONAL
C     PATTERN.  EACH PATTERN IS CLASSIFIED INTO ONE OF TWO CLASSES
C     BY MEANS OF A BAYES CLASSIFIER.  A STOCHASTIC APPROXIMATION
C     TECHNIQUE (CF) IS USED IN THE ESTIMATION OF THE MEAN OF THE TWO CLASS
C     SIGNATURES.  UPDATED ESTIMATES OF COVARIANCE MATRICES ARE
C     MADE FOR BLOCKS OF CLASSIFIED DATA.  THE A PRIORI PROBABILITIES
C     ASSOCIATED WITH THE TWO CLASSES ARE FURTHER ASSUMED TO BE
C     EQUIVALENT.  A BOUNDARY IS DEFINED SEPARATING THE TWO CLASSES
C     AND IS STORED IN DISK MEMORY IN PROPER FORM FOR DISPLAY
C     VIA THE DATA DISK VIDEO SYSTEM.
C
C
C
C     VARIABLES:
C
C     NC     -NUMBER OF CLASSES
C     IC     -CLASS CONSIDERATION INTERVAL
C     IX     -HORIZONTAL PICTURE ARRAY INDEX
C     IY     -VERTICAL PICTURE ARRAY INDEX
C     ICLASS -EITHER 1 OR 2 INDICATING WHETHER LAST PATTERN
C             WAS CLASSIFIED A MEMBER OF CLASS ONE OR TWO
C     INDEX  -AN INDEX OF THE IADATA ARRAY
C
```

```
C     ARRAYS:
C
C     DATAI  -ARRAY TO RECEIVE 128 FOUR DIMENSIONAL DATA POINTS
C             AS INPUT
C     C1     -ARRAY CONTAINING ESTIMATE OF CLASS ONE COVARIANCE
C             MATRIX
C     C2     -ARRAY CONTAINING ESTIMATE OF CLASS TWO COVARIANCE
C             MATRIX
C     C1I    -ARRAY CONTAINING INVERSE OF CLASS ONE COVARIANCE
C     C2I    -ARRAY CONTAINING INVERSE OF CLASS TWO COVARIANCE
C     U1EST  -ARRAY CONTAINING INITIAL ESTIMATE OF CLASS ONE MEAN
C             VECTOR
C     U2EST  -ARRAY CONTAINING INITIAL ESTIMATE OF CLASS TWO MEAN
C             VECTOR
C     UA     -ARRAY CONTAINING AN INTERMEDIATE ESTIMATE OF CLASS ONE
C             MEAN VECTOR
C     UB     -ARRAY CONTAINING AN INTERMEDIATE ESTIMATE OF CLASS TWO
C             MEAN VECTOR
C     U1     -ARRAY CONTAINING A PROJECTED ESTIMATE OF CLASS ONE
C             MEAN VECTOR
C     U2     -ARRAY CONTAINING A PROJECTED ESTIMATE OF CLASS TWO
C             MEAN VECTOR
C     X      -ARRAY CONTAINING AN INDIVIDUAL PATTERN VECTOR
C     U1BAR  -ARRAY CONTAINING DIVERGENCE INFORMATION CONCERNING CLASS
C             ONE MEAN ESTIMATES
C     U2BAR  -ARRAY CONTAINING DIVERGENCE INFORMATION CONCERNING CLASS
C             TWO MEAN ESTIMATES
C     TIME1  -ARRAY CONTAINING TIME INFORMATION FOR USE IN STOCHASTIC
C             APPROXIMATION OF CLASS ONE MEAN VECTOR
C     TIME2  -ARRAY CONTAINING TIME INFORMATION FOR USE IN STOCHASTIC
C             APPROXIMATION OF CLASS TWO MEAN VECTOR
C     CIA    -ARRAY CONTAINING CONVERGENCE INFORMATION FOR USE IN
C             STOCHASTIC APPROXIMATION OF CLASS ONE MEAN
C     CIB    -ARRAY CONTAINING CONVERGENCE INFORMATION FOR USE IN
C             STOCHASTIC APPROXIMATION OF CLASS TWO MEAN
```

```
C      IADATA -ARRAY CONTAINING ROW BOUNDARY POSITIONS
C      ICDATA -A LOGICAL STAR*1 ARRAY USED IN BUILDING A DISK OUTPUT
C              FILE FOR DISPLAY VIA DATA-DISK VIDEO SYSTEM
C      IODATA -ARRAY CONTAINING TRUE BOUNDARY
C
C      SUBROUTINES:
C
C      INPUT     -READS DATA FROM DISK INPUT FILE
C      DECIDE    -BAYES CLASSIFIER FOR TWO CLASSES
C      BOUND     -SUBROUTINE TO PLACE CLASS BOUNDARY
C      AVERAGE   -COMPUTES AVERAGES USED AS INDICATIONS OF DIVERGENCE
C                 OF STOCHASTIC APPROXIMATIONS
C      INTERVAL  -SUBROUTINE TO CALCULATE CONFIDENCE INTERVALS FOR
C                 STOCHASTIC APPROXIMATIONS
C      PROJECT   -STOCHASTIC APPROXIMATION SUBROUTINE
C      PROCOV    -MAKES ESTIMATES OF COVARIANCE MATRIX
C      COVAR     -COMPUTES COVARIANCE OF DATA SAMPLES IN RECURSIVE
C                 FORM
C      MINV      -INVERTS A MATRIX
C      MPRD      -FORMS THE PRODUCT OF TWO MATRICES
C      OUTPUT    -WRITES BOUNDARY DATA TO A DISK FILE FOR DISPLAY
C                 PURPOSES
C
C
C
       DIMENSION DATAI(512),C1(16),C2(16),C1I(16),C2I(16),
      1U1EST(4),U2EST(4),UA(4),UB(4),U1(4),U2(4),X(4),
      1U1BAR(4),U2BAR(4),TIME1(4),TIME2(4),CIA(4),CIB(4)
       DIMENSION IADATA(128),IODATA(128)
       EQUIVALENCE (INT,DUM)
       LOGICAL*1 ICDATA(128),DUM(2)
       COMMON /DUMB1/DATAI
       COMMON /DUMB2/ICDATA
```

```
      DATA IAST/'*'/
      DATA IBLNK/' '/
      WRITE(7,10)
   10 FORMAT(' SPECIFY INPUT FILE'/)
      CALL ASSIGN(1,'SYO:ABCDEF.DAT',-14,'RDO','NC',1)
      DEFINE FILE 1(128,1024,U,I1)
      WRITE(7,20)
   20 FORMAT(' SPECIFY OUTPUT FILE'/)
      CALL ASSIGN(2,'SYO:ABCDEF.DAT',-14,'NEW','NC',1)
      DEFINE FILE 2(128,64,U,I2)
      WRITE(7,25)
   25 FORMAT(' SPECIFY CLASS CONSIDERATION INTERVAL')
      READ(7,26)IC
   26 FORMAT(I3)
      WRITE(7,30)
   30 FORMAT(' SPECIFY 4X4 EST. OF CLASS 1 COV. MATRIX')
      DO 35 NM=1,4
      READS(5,40)(C1(I),I=NM,NM+12,4)
   35 CONTINUE
      WRITE(7,36)
   36 FORMAT(' SPECIFY 4X4 EST. OF CLASS 2 COV. MATRIX')
      DO 37 NM=1,4
      READ(5,40)(C2(I),I=NM,NM+12,4)
   37 CONTINUE
   40 FORMAT(4F10.5)
      WRITE(7,50)
   50 FORMAT(' SPECIFY EST. OF CLASS 1 MEAN VECTOR')
      READ(5,40)(U1EST(I),I=1,4)
      WRITE(7,60)
   60 FORMAT(' SPECIFY EST. OF CLASS 2 MEAN VECTOR')
      READ(5,40)(U2EST(I),I=1,4)
      NC=2
      CALL PROCOV(C1,C2,C1I,C2I,D1,D2,X,ICLASS,IC,0)
```

```
C
C BEGINNING OF BOUNDARY DEFINITION PROCEEDURE FOR THE 128 X 128
C PICTURE
C
      DO 70 IY=1,128
      INDEX=1
      DO 71 M=1,128
      IADATA(M)=129
   71 CONTINUE
      DO 75 NM=1,4
      UA(NM)=U1EST(NM)
      UB(NM)=U2EST(NM)
      U1(NM)=U1EST(NM)
      U2(NM)=U2EST(NM)
      TIME1(NM)=0.
      TIME2(NM)=0.
   75 CONTINUE
C
C READ A ROW AND PLACE BOUNDARIES WHERE NECESSARY
C
      CALL INPUT(IY)
      DO 90 J=1,512,4
      IX=(J+3)/4
      X(1)=DATAI(J)
      X(2)=DATAI(J+1)
      X(3)=DATAI(J+2)
      X(4)=DATAI(J+3)
      CALL DECIDE(X,C1I,C2I,U1,U2,D1,D2,ICLASS)
      CALL BOUND(IADATA,ICLASS,IX,IC,INDEX)
      CALL AVERAGE(X,U1,U2,U1BAR,U2BAR,ICLASS,TIME1,TIME2)
      CALL INTERVAL(TIME1,TIME2,ICLASS,CIA,CIB)
      CI=FLOAT(IC)
      IF(ICLASS.EQ.2)GO TO 84
      IF(TIME1(1).LE.CI)GO TO 80
```

```
      IF(ABS(U1BAR(1)).LE.CIA(1))GO TO 80
      TIME1(1)=0.
      UA(1)=U1(1)
   80 IF(TIME1(2).LE.CI)GO TO 81
      IF(ABS(U1BAR(2)).LE.CIA(2))GO TO 81
      TIME1(2)=0.
      UA(2)=U1(2)
   81 IF(TIME1(3).LE.CI)GO TO 82
      IF(ABS(U1BAR(3)).LE.CIA(3))GO TO 82
      TIME1(3)=0.
      UA(3)=U1(3)
   82 IF(TIME1(4).LE.CI)GO TO 88
      IF(ABS(U1BAR(4)).LE.CIA(4))GO TO 88
      TIME1(4)=0.
      UA(4)=U1(4)
      GO TO 88
   84 IF(TIME2(1).LE.CI)GO TO 85
      IF(ABS(U2BAR(1)).LE.CIB(1))GO TO 85
      TIME2(1)=0.
      UB(1)=U2(1)
   85 IF(TIME2(2).LE.CI)GO TO 86
      IF(ABS(U2BAR(2)).LE.CIB(2))GO TO 86
      TIME2(2)=0.
      UB(2)=U2(2)
   86 IF(TIME2(3).LE.CI)GO TO 87
      IF(ABS(U2BAR(3)).LE.CIB(3))GO TO 87
      TIME2(3)=0.
      UB(3)=U2(3)
   87 IF(TIME2(4).LE.CI)GO TO 88
      IF(ABS(U2BAR(4)).LE.CIB(4))GO TO 38
      TIME2(4)=0.
      UB(4)=U2(4)
   88 CALL PROJECT(X,U1,U2,ICLASS,UA,UB,TIME1,TIME2)
      CALL PROCOV(C1,C2,C1I,C2I,D1,D2,X,ICLASS,IC,1)
```

```
   90 CONTINUE
      DO 91 M=1,128
      IODATA(M)=IBLNK
   91 CONTINUE
      M=1
   92 IAM=IADATA(M)
      IF(IAM.GT.128)GO TO 93
      IODATA(IAM)=IAST
      M=M+1
      GO TO 92
   93 CONTINUE
      DO 100 I=1,128
      IF(IODATA(I).EQ.IBLNK)IODATA(I)=0
      IF(IODATA(I).EQ.IAST)IODATA(I)=129
      INT=IODATA(I)
      ICDATA(I)=DUM(1)
  100 CONTINUE
      CALL OUTPUT(IY)
   70 CONTINUE
      WRITE(7,110)
  110 FORMAT(' CLASSIFICATION COMPLETE')
      ENDFILE 1
      ENDFILE 2
      STOP
      END
```

```
      SUBROUTINE INPUT (IY)
C THIS SUBROUTINE READS ONE ROW OF FOUR-SIMENSIONAL DATA OF A 128 X 128
C DATA POINT ARRAY FROM A PRESPECIFIED DISK FILE.  THE ROW OF DATA
C IS READ INTO THE ARRAY DATAI.
      DIMENSION DATAI(512)
      COMMON /DUMB1/DATAI
      READ(1'IY)DATAI
      RETURN
      END
```

```
      SUBROUTINE DECIDF(X,CAI,CBI,UA,UB,DA,DB,ICLASS)
C THIS SUBROUTINE IMPLEMENTS A BAYES CLASSIFIER FOR TWO CLASSES
C OF EQUAL A PRIORI PROBABILITY.
      DIMENSION X(4),CAI(16,),CBI(16),UA(4),UB(4),
     1YA(4),YB(4),RA(4),RB(4),A(1),B(1)
      DO 10 I=1,4
      YA(I)=X(I)-UA(I)
      YB(I)=X(I)-UB(I)
   10 CONTINUE
      CALL MPRD(YA,CAI,RA,1,4,4)
      CALL MPRD(YB,CBI,RB,1,4,4)
      CALL MPRD(RA,YA,A,1,4,1)
      CALL MPRD(RB,YB,B,1,4,1)
      F1=-(ALOG(DA)+A(1))
      F2=-(ALOG(DB)+B(1))
      ICLASS=1
      IF(F2.GT.F1)ICLASS=2
      RETURN
      END
```

```
      SUBROUTINE BOUND(IDATA,ICLASS,IX,IC,INDEX)
C THIS SUBROUTINE FORMS A BOUNDARY BETWEEN THE TWO DATA CLASSES.
      DIMENSION IDATA(128),ICON(32)
      IF(IX.GT.1)GO TO 10
      NCLASS=ICLASS
   10 DO 20 N=1,IC-1
      ICON(N)=ICON(N+1)
   20 CONTINUE
      ICON(IC)=ICLASS
      IF(IX.LT.IC)GO TO 30
      ICNT1=0
      ICNT2=0
      DO 40 II=1,IC
      IF(ICON(II).EQ.1)ICNT1=ICNT1+1
      IF(ICON(II).EQ.2)ICNT2=ICNT2+1
   40 CONTINUE
      IF(ICNT1.GT.ICNT2)MCLASS=1
      IF(ICNT2.GT.INCT1)MCLASS=2
      IF(MCLASS.EQ.NCLASS)GO TO 30
      NCLASS=MCLASS
      IDATA(INDEX)=IX-IC/2
      INDEX=INDEX+1
   30 RETURN
      END
```

```
      SUBROUTINE AVERAGE(X,U1,U2,X1BAR,X2BAR,ICLASS,TIMEX,TIMEY)
C THIS SUBROUTINE KEEPS AN INDEPENDENT RUNNING TIME AVERAGE OF THE
C DEVIATION OF THE DATA FROM THE TRUE OR PROJECTED MEAN OF EACH
C CLASS.
      DIMENSION X(4),U1(4),U2(4),X1BAR(4),X2BAR(4),TIMEX(4),TIMEY(4)
      IF(ICLASS.EQ.2)GO TO 20
      DO 10 I=1,4
      X1BAR(I)=(1./(TIMEX(I)+1.))*(TIMEX(I)*X1BAR(I)+(X(I)-U1(I)))
   10 CONTINUE
      GO TO 30
   20 DO 30 I=1,4
      X2BAR(I)=(1./(TIMEY(I)+1.))*(TIMEY(I)*X2BAR(I)+(X(I)-U2(I)))
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTERVAL(TIMEX,TIMEY,ICLASS,CIA,CIB)
C THIS SUBROUTINE CALCULATES THE CONFIDENCE INTERVAL ASSOCIATED WITH
C EACH COMPONENT OF THE MEAN VECTOR FOR THE CLASS OF INTEREST.
C THE CONFIDENCE INTERVAL IS FOR USE AS AN INDICATION OF DIVERGENCE.
      DIMENSION TIMEX(4),TIMEY(4),CIA(4),CIB(4)
      IF(ICLASS.EQ.2)GO TO 20
      IF(TIMEX(1).EQ.0.)GO TO 10
      CIA(1)=3./SQRT(TIMEX(1))
   10 IF(TIMEX(2).EQ.0.)GO TO 12
      CIA(2)=3./SQRT(TIMEX(2))
   12 IF(TIMEX(3).EQ.0.)GO TO 14
      CIA(3)=3./SQRT(TIMEX(3))
   14 IF(TIMEX(4).EQ.0.)GO TO 40
      CIA(4)=3./SQRT(TIMEX(4))
      GO TO 40
   20 IF(TIMEY(1).EQ.0.)GO TO 30
      CIB(1)=3./SQRT(TIMEY(1))
   30 IF(TIMEY(2).EQ.0.)GO TO 32
      CIB(2)=3./SQRT(TIMEY(2))
   32 IF(TIMEY(3).EQ.0.)GO TO 34
      CIB(3)=3./SQRT(TIMEY(3))
   34 IF(TIMEY(4).EQ.0.)GO TO 40
      CIB(4)=3./SQRT(TIMEY(4))
   40 RETURN
      END
```

```
      SUBROUTINE PROJECT(X,UX,UY,ICLASS,UA,UB,TIMEX,TIMEY)
C THIS SUBROUTINE PROJECTS EACH COMPONENT OF THE MEAN VECTOR OF THE
C CLASS OF INTEREST.  THE PROJECTION IS MADE USING A STOCHASTIC
C APPROXIMATION.  THE TWO COMPONENTS ARE PROJECTED INDEPENDENTLY OF
C ONE ANOTHER.  (CHIEN AND FU)
      DIMENSION X(4),UX(4),UY(4),UA(4),UB(4),TIMEX(4),TIMEY(4)
      IF(ICLASS.EQ.2)GO TO 20
      DO 10 I=1,4
      UX(I)=UX(I)-UA(I)
      X(I)=X(I)-UA(I)
      USTAR=(1.+1./(TIMEX(I)+1.))*UX(I)
      GAMMA=6.*(TIMEX(I)+2.)/((TIMEX(I)+3.)*(2.*(TIMEX(I)+1.)+3.))
      UX(I)=USTAR+GAMMA*(X(I)-USTAR)
      TIMEX(I)=TIMEX(I)+1.
      UX(I)=UX(I)+UA(I)
   10 CONTINUE
      GO TO 30
   20 DO 30 I=1,4
      UY(I)=UY(I)-UB(I)
      X(I)=X(I)-UB(I)
      USTAR=(1.+1./(TIMEY(I)+1.))*UY(I)
      GAMMA=6.*(TIMEY(I)+2.)/((TIMEY(I)+3.)*(2.*(TIMEY(I)+1.)+3.))
      UY(I)=USTAR+GAMMA*(X(I)-USTAR)
      TIMEY(I)=TIMEY(I)+1.
      UY(I)=UY(I)+UB(I)
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PROCOV(CA,CB,CAINV,CBINV,DA,DB,X,ICLASS,IC,IFLAG)
C THIS SUBROUTINE MAKES A NEW ESTIMATE OF THE COVARIANCE OF A
C CLASS AFTER IC POINTS HAVE BEEN CLASSIFIED INTO THAT PARTICULAR
C CLASS.
      DIMENSION CA(16),CB(16),CAINV(16),CBINV(16),
     1PHIA(16),PHIB(16),X(4)
      IF(IFLAG.GT.0)GO TO 10
      ICNTA=1
      ICNTB=1
      DO 5 I=1,16
      CAINV(I)=CA(I)
      CBINV(I)=CB(I)
    5 CONTINUE
      CALL MINV(CAINV,4,DA)
      CALL MINV(CBINV,4,DB)
      GO TO 50
   10 IF(ICLASS.EQ.2)GO TO 30
      CALL COVAR(PHIA,X,ICNTA,4)
      ICNTA=ICNTA+1
      IF(ICNTA.LE.IC)GO TO 50
      DO 20 I=1,16
      CA(I)=PHIA(I)
      CAINV(I)=PHIA(I)
   20 CONTINUE
      CALL MINV(CAINV,4,DA)
      ICNTA=1
      GO TO 50
   30 CALL COVAR(PHIB,X,ICNTB,4)
      ICNTB=ICNTB+1
      IF(ICNTB.LE.IC)GO TO 50
      DO 40 I=1,16
      CB(I)=PHIB(I)
      CBINV(I)=PHIB(I)
```

```
   40 CONTINUE
      CALL MINV(CBINV,4,DB)
      ICNTB=1
   50 RETURN
      END
```

```
      SUBROUTINE COVAR(COV,X,INDEX,NDIM)
C THIS SUBROUTINE IMPLEMENTS THE RECURSIVE FORM OF COVARIANCE
C ESTIMATION.
      DIMENSION COV(4,4),X(4),XM(4)
      X1=FLOAT(INDEX)
      XO=X1-1.
      IF(INDEX.NE.1)GO TO 5
      DO 4 I=1,NDIM
      XM(I)=X(I)
      DO 3 J=1,NDIM
      COV(I,J)=0.
    3 CONTINUE
    4 CONTINUE
      GO TO 20
    5 DO 10 I=1,NDIM
      DO 11 J=1,NDIM
      COV(I,J)=(1./X1)*(XO*COV(I,J)+XO*XM(I)*XM(J)+X(I)*X(J))
     1-(1./(X1*X1))*XO*XM(I)+X(I))*(XO*XM(J)+X(J))
   11 CONTINUE
   10 CONTINUE
      DO 15 I=1,NDIM
   15 XM(I)=(1./X1)*(XO*XM(I)+X(I))
   20 RETURN
      END
```

```
0001         SUBROUTINE MINV(A,N,D)
      C THIS SUBROUTINE FINDS THE INVERSE OF A GENERAL MATRIX
      C 'A', WHICH IS DESTROYED IN COMPUTATION.  THE INVERSE
      C IS RETURNED IN 'A'.  THE DETERMINANT IS CALCULATED,
      C 'D'.  'N' IS THE ORDER OF THE MATRIX, 'L' IS A WORK
      C VECTOR OF LENGTH 'N', AND 'M' IS A WORK VECTOR OF
      C LENGTH 'N'.
0002         DIMENSION A(16),L(4),M(4)
0003         D=1.0
0004         NK=-N
0005         DO 80 K=1,N
0006         NK=NK+N
0007         L(K)=K
0008         M(K)=K
0009         KK=NK+K
0010         BIGA=A(KK)
0011         DO 20 J=K,N
0012         IZ=N*(J-1)
0013         DO 20 I=K,N
0014         IJ=IZ+I
0015      10 IF (ABS(BIGA)-ABS(A(IJ))) 15,20,20
0016      15 BIGA=A(IJ)
0017         L(K)=I
0018         M(K)=J
0019      20 CONTINUE
0020         J=L(K)
0021         IF(J-K) 35,35,25
0022      25 KI=K-N
0023         DO 30 I=1,N
0024         KI=KI+N
0025         HOLD=-A(KI)
0026         JI=KI-K+J
0027         A(KI)=A(JI)
0028      30 A(JI)=HOLD
```

```
35 I=M(K)
   IF(I-K)45,45,38
38 JP=N*(I-1)
   DO 40 J=1,N
   JK=NK+J
   JI=JP+J
   HOLD=-A(JK)
   A(JK)=A(JI)
40 A(JI)=HOLD
45 IF(BIGA) 48,46,48
46 D=0.
   RETURN
48 DO 55 I=1,N
   IF(I-K) 50,55,50
50 IK=NK+I
   A(IK)=A(IK)/(-BIGA)
55 CONTINUE
   DO 65 I=1,N
   IK=NK+I
   HOLD=A(IK)
   IJ=I-N
   DO 65 J=1,N
   IJ=IJ+N
   IF(I-K) 60,65,60
60 IF(J-K) 62,65,62
62 KJ=IJ-I+K
   A(IJ)=HOLD*A(KJ)+A(IJ)
65 CONTINUE
   KJ=K-N
   DO 75 J=1,N
   KJ=KJ+N
   IF(J-K) 70,75,70
70 A(KJ)=A(KJ)/BIGA
75 CONTINUE
```

```
      D=D*BIGA
      A(KK)=1.0/BIGA
   80 CONTINUE
      K=N
  100 K=K-1
      IF(K) 150,150,105
  105 I=L(K)
      IF(I-K) 120,120,108
  108 JQ=N*(K-1)
      JR=N*(I-1)
      DO 110 J=1,N
      JK=JQ+J
      HOLD=A(JK)
      JI=JR+J
      A(JK)=-A(JI)
  110 A(JI)=HOLD
  120 J=M(K)
      IF(J-K) 100,100,125
  125 KI=K-N
      DO 130 I=1,N
      KI=KI+N
      HOLD=A(KI)
      JI=KI-K+J
      A(KI)=-A(JI)
  130 A(JI)=HOLD
      GO TO 100
  150 RETURN
      END
```

```
      SUBROUTINE MPRD(A,B,R,N,M,L)
C THIS SUBROUTINE FORMS THE PRODUCT OF TWO GENERAL MATRICES
C 'A' AND 'B', AND RETURNS THE PRODUCT IN 'R'.  'N' IS THE NUMBER
C OF ROWS IN 'A', 'M' IS THE NUMBER OF COLUMNS IN 'A' AND ROWS
C IN 'B', AND 'L' IS THE NUMBER OF COLUMNS IN 'B'.
      DIMENSION A(16),B(16),R(16)
      IR=0
      IK=-M
      DO 10 K=1,L
      IK=IK+M
      DO 10 J=1,N
      IR=IR+1
      JI=J-N
      IB=IK
      R(IR)=0
      DO 10 I=1,M
      JI=JI+N
      IB=IB+1
   10 R(IR)=R(IR)+A(JI)*B(IB)
      RETURN
      END
```

C-2

```
0001        SUBROUTINE OUTPUT (IY)
      C THIS SUBROUTINE WRITES ONE 128 POINT ROW (BYTE DATA POINTS) OF A
      C 128 X 128 POINT ARRAY FOR BOUNDARY DISPLAY VIA VIDEO DISPLAY SYSTEM.
0002        LOGICAL*1 IODATA(128)
0003        COMMON /DUMB2/ICDATA
0004        WRITE(2'IY)ICDATA
0005        RETURN
0006        END
```

APPENDIX D

COMPILED FORTRAN IV PROGRAM LISTING OF ADAPTIVE BAYES CLASSIFIER

INCORPORATING SECOND DEGREE PF ALGORITHM

```
C
C
C     THIS PROGRAM READS A 128 X 128 DATA POINT ARRAY FROM A DISK
C     FILE, EACH POINT OF WHICH MAY BE CONSIDERED A FOUR DIMENSIONAL
C     PATTERN.  EACH PATTERN IS CLASSIFIED INTO ONE OF TWO CLASSES
C     BY MEANS OF A BAYES CLASSIFIER.  A STOCHASTIC APPROXIMATION
C     TECHNIQUE (SECOND DEGREE PF) IS USED IN THE ESTIMATION OF THE
C     MEAN OF THE TWO CLASS SIGNATURES.  UPDATED ESTIMATES OF COVARIANCE
C     MATRICES ARE MADE FOR BLOCKS OF CLASSIFIED DATA.  THE A PRIORI
C     PROBABILITIES ASSOCIATED WITH THE TWO CLASSES ARE FURTHER ASSUMED
C     TO BE EQUIVALENT.  A BOUNDARY IS DEFINED SEPARATING THE TWO CLASSES
C     AND IS STORED IN DISK MEMORY IN PROPER FORM FOR DISPLAY
C     VIA THE DATA-DISK VIDEO SYSTEM.
C
C
C
C     VARIABLES:
C
C     NC     -NUMBER OF CLASSES
C     IC     -CLASS CONSIDERATION INTERVAL
C     IX     -HORIZONTAL PICTURE ARRAY INDEX
C     IY     -VERTICAL PICTURE ARRAY INDEX
C     ICLASS -EITHER 1 OR 2 INDICATING WHETHER LAST PATTERN
C              WAS CLASSIFIED A MEMBER OF CLASS ONE OR TWO
C     INDEX  -AN INDEX OF THE IADATA ARRAY
C
```

```
C     ARRAYS:
C
C     DATAI  -ARRAY TO RECEIVE 128 FOUR DIMENSIONAL DATA POINTS
C             AS INPUT
C     C1     -ARRAY CONTAINING ESTIMATE OF CLASS ONE COVARIANCE
C             MATRIX
C     C2     -ARRAY CONTAINING ESTIMATE OF CLASS TWO COVARIANCE
C             MATRIX
C     C1I    -ARRAY CONTAINING INVERSE OF CLASS ONE COVARIANCE
C     C2I    -ARRAY CONTAINING INVERSE OF CLASS TWO COVARIANCE
C     U1EST  -ARRAY CONTAINING INITIAL ESTIMATE OF CLASS ONE MEAN
C             VECTOR
C     U2EST  -ARRAY CONTAINING INITIAL ESTIMATE OF CLASS TWO MEAN
C             VECTOR
C     UA     -ARRAY CONTAINING AN INTERMEDIATE ESTIMATE OF CLASS ONE
C             MEAN VECTOR
C     UB     -ARRAY CONTAINING AN INTERMEDIATE ESTIMATE OF CLASS TWO
C             MEAN VECTOR
C     U1     -ARRAY CONTAINING A PROJECTED ESTIMATE OF CLASS ONE
C             MEAN VECTOR
C     U2     -ARRAY CONTAINING A PROJECTED ESTIMATE OF CLASS TWO
C             MEAN VECTOR
C     X      -ARRAY CONTAINING AN INDIVIDUAL PATTERN VECTOR
C     U1BAR  -ARRAY CONTAINING DIVERGENCE INFORMATION CONCERNING CLASS
C             ONE MEAN ESTIMATES
C     U2BAR  -ARRAY CONTAINING DIVERGENCE INFORMATION CONCERNING CLASS
C             TWO MEAN ESTIMATES
C     TIME1  -ARRAY CONTAINING TIME INFORMATION FOR USE IN STOCHASTIC
C             APPROXIMATION OF CLASS ONE MEAN VECTOR
C     TIME2  -ARRAY CONTAINING TIME INFORMATION FOR USE IN STOCHASTIC
C             APPROXIMATION OF CLASS TWO MEAN VECTOR
C     CIA    -ARRAY CONTAINING CONVERGENCE INFORMATION FOR USE IN
C             STOCHASTIC APPROXIMATION OF CLASS ONE MEAN
```

```
      C      CIB    -ARRAY CONTAINING CONVERGENCE INFORMATION FOR USE IN
      C              STOCHASTIC APPROXIMATION OF CLASS TWO MEAN
      C      E1     -ARRAY CONTAINING AN ESTIMATE OF THE MEAN SQUARE ERROR
      C              OF THE CLASS ONE MEAN VECTOR
      C      E2     -ARRAY CONTAINING AN ESTIMATE OF THE MEAN SQUARE ERROR
      C              OF THE CLASS TWO MEAN VECTOR
      C      IADATA -ARRAY CONTAINING ROW BOUNDARY POSITIONS
      C      ICDATA -A LOGICAL STAR*1 ARRAY USED IN BUILDING A DISK OUTPUT
      C              FILE FOR DISPLAY VIA DATA-DISK VIDEO SYSTEM
      C      IODATA -ARRAY CONTAINING TRUE BOUNDARY
      C
      C      SUBROUTINES:
      C
      C      INPUT      -READS DATA FROM DISK INPUT FILE
      C      DECIDE     -BAYES CLASSIFIER FOR TWO CLASSES
      C      BOUND      -SUBROUTINE TO PLACE CLASS BOUNDARY
      C      PROJECT    -STOCHASTIC APPROXIMATION SUBROUTINE
      C      PROCOV     -MAKES ESTIMATES OF COVARIANCE MATRIX
      C      COVAR      -COMPUTES COVARIANCE OF DATA SAMPLES IN RECURSIVE
      C                  FORM
      C      MINV       -INVERTS A MATRIX
      C      MPRD       -FORMS THE PRODUCT OF TWO MATRICES
      C      OUTPUT     -WRITES BOUNDARY DATA TO A DISK FILE FOR DISPLAY
      C                  PURPOSES
      C
      C
      C
0001          DIMENSION DATAI(512),C1(16),C2(16),C1I(16),C2I(16),
             1U1EST(4),U2EST(4),UA(4),UB(4),U1(4),U2(4),X(4),
             1U1BAR(4),U2BAR(4),TIME1(4),TIME2(4),CIA(4),CIB(4),
             1E1(4),E2(4)
0002          DIMENSION IADATA(128),IODATA(128)
0003          EQUIVALENCE (INT,DUM)
0004          LOGICAL*1 ICDATA(128),DUM(2)
```

```
      COMMON /DUMB1/DATAI
      COMMON /DUMB2/ICDATA
      DATA IAST/'*'/
      DATA IBLNK/' '/
      WRITE(7,10)
   10 FORMAT(' SPECIFY INPUT FILE'/)
      CALL ASSIGN(1,'SY0:ABSDEF.DAT',-14,'RDO','NC',1)
      DEFINE FILE 1(128,1024,U,I1)
      WRITE(7,20)
   20 FORMAT(' SPECIFY OUTPUT FILE'/)
      CALL ASSIGN(2,'SY0:ABCDEF.DAT',-14,'NEW','NC',1)
      DEFINE FILE 2(128,64,U,I2)
      WRITE(7,25)
   25 FORMAT(' SPECIFY CLASS CONSIDERATION INTERVAL')
      READ(7,26)IC
   26 FORMAT(I3)
      WRITE(7,30)
   30 FORMAT(' SPECIFY 4X4 EST. OF CLASS 1 COV. MATRIX')
      DO 35 NM=1,4
      READ(5,40)(C1(I),I=NM,NM+12,4)
   35 CONTINUE
      WRITE(7,36)
   36 FORMAT(' SPECIFY 4X4 EST. OF CLASS 2 COV. MATRIX')
      DO 37 NM=1,4
      READ(5,40)(C2(I),I=NM,NM+12,4)
   37 CONTINUE
   40 FORMAT(4F10.5)
      WRITE(7,50)
   50 FORMAT(' SPECIFY EST. OF CLASS 1 MEAN VECTOR')
      READ(5,40)(U1EST(I),I=1,4)
      WRITE(7,60)
   60 FORMAT(' SPECIFY EST. OF CLASS 2 MEAN VECTOR')
      READ(5,40)(U2EST(I),I=1,4)
      NC=2
```

```
      CALL PROCOV(C1,C2,C1I,C2I,D1,D2,X,ICLASS,IC,0)
C
C BEGINNING OF BOUNDARY DEFINITION PROCEDURE FOR THE 128 X 128
C PICTURE
C
      DO 70 IY=1,128
      INDEX=1
      DO 71 M=1,128
      IADATA(M)=129
   71 CONTINUE
      DO 75 NM=1,4
      UA(NM)=U1EST(NM)
      UB(NM)=U2EST(NM)
      U1(NM)=U1EST(NM)
      U2(NM)=U2EST(NM)
      TIME1(NM)=0.
      TIME2(NM)=0.
      E1(NM)=0.
      E2(NM)=0.
   75 CONTINUE
C
C READ A ROW AND PLACE BOUNDARIES WHERE NECESSARY
C
      CALL INPUT(IY)
      DO 90 J=1,512,4
      IX=(J+3)/4
      X(1)=DATAI(J)
      X(2)=DATAI(J+1)
      X(3)=DATAI(J+2)
      X(4)=DATAI(J+3)
      CALL DECIDE(X,C1I,C2I,U1,U2,D1,D2,ICLASS)
      CALL BOUND(IADATA,ICLASS,IX,IC,INDEX)
   88 CALL PROJECT(X,U1,U2,ICLASS,E1,E2,TIME1,TIME2)
```

```
      CALL PROCOV(C1,C2,C1I,C2I,D1,D2,X,ICLASS,IC,1)
   90 CONTINUE
      DO 91 M=1,128
      IODATA(M)=IBLNK
   91 CONTINUE
      M=1
   92 IAM=IADATA(M)
      IF(IAM.GT.128)GO TO 93
      IODATA(IAM)=IAST
      M=M+1
      GO TO 92
   93 CONTINUE
      DO 100 I=1,128
      IF(IODATA(I).EQ.IBLANK)IODATA(I)=0
      IF(IODATA(I).EQ.IAST)IODATA(1)=129
      INT=IODATA(I)
      ICDATA(I)=DUM(1)
  100 CONTINUE
      CALL OUTPUT(IY)
   70 CONTINUE
      WRITE(7,110)
  110 FORMAT(' CLASSIFICATION COMPLETE')
      ENDFILE 1
      ENDFILE 2
      STOP
      END
```

```
0001        SUBROUTINE INPUT (IY)
     C THIS SUBROUTINE READS ONE ROW OF FOUR-DIMENSIONAL DATA OF A 128 X 128
     C DATA POINT ARRAY FROM A PRESPECIFIED DISK FILE.  THE ROW OF DATA
     C IS READ INTO THE ARRAY DATAI.
0002        DIMENSION DATAI(512)
0003        COMMON /DUMB1/DATAI
0004        READ(1'IY)DATAI
0005        RETURN
0006        END
```

```
      SUBROUTINE DECIDE(X,CAI,CBI,UA,UB,DA,DB,ICLASS)
C THIS SUBROUTINE IMPLEMENTS A BAYES CLASSIFIER FOR TWO CLASSES
C OF EQUAL A PRIORI PROBABILITY.
      DIMENSION X(4),CAI(16),CBI(16),UA(4),UB(4),
     1YA(4),YB(4),RA(4),RB(4),A(1),B(1)
      DO 10 I=1,4
      YA(I)=X(I)-UA(I)
      YB(I)=X(I)-UB(I)
   10 CONTINUE
      CALL MPRD(YA,CAI,RA,1,4,4)
      CALL MPRD(YB,CBI,RB,1,4,4)
      CALL MPRD(RA,YA,A,1,4,1)
      CALL MPRD(RB,YB,B,1,4,1)
      F1=-(ALOG(DA)+A(1))
      F2=-(ALOG(DB)+B(1))
      ICLASS=1
      IF(F2.GT.F1)ICLASS=2
      RETURN
      END
```

```
      SUBROUTINE BOUND(IDATA,ICLASS,IX,IC,INDEX)
C THIS SUBROUTINE FORMS A BOUNDARY BETWEEN THE TWO DATA CLASSES.
      DIMENSION IDATA(128),ICON(32)
      IF(IX.GT.1)GO TO 10
      NCLASS=ICLASS
   10 DO 20 N=1,IC-1
      ICON(N)=ICON(N+1)
   20 CONTINUE
      ICON(IC)=ICLASS
      IF(IX.LT.IC)GO TO 30
      ICNT1=0
      ICNT2=0
      DO 40 II=1,IC
      IF(ICON(II).EQ.1)ICNT1=ICNT1+1
      IF(ICON(II).EQ.2)ICNT2=ICNT2+1
   40 CONTINUE
      IF(ICNT1.GT.ICNT2)MCLASS=1
      IF(ICNT2.GT.ICNT1)MCLASS=2
      IF(MCLASS.EQ.NCLASS)GO TO 30
      NCLASS=MCLASS
      IDATA(INDEX)=IX-IC/2
      INDEX=INDEX+1
   30 RETURN
      END
```

```
      SUBROUTINE PROJECT(X,UX,UY,ICLASS,E1,E2,TIMEX,TIMEY)
C THIS SUBROUTINE PROJECTS EACH COMPONENT OF THE MEAN VECTOR OF THE
C CLASS OF INTEREST.  THE PROJECTION IS MADE USING A STOCHASTIC
C APPROXIMATION.  THE FOUR COMPONENTS ARE PROJECTED INDEPENDENTLY OF
C ONE ANOTHER.  (SECOND DEGREE POLYNOMIAL FIT)
      DIMENSION X(4),UX(4),UY(4),E1(4),E2(4),TIMEX(4),TIMEY(4),
     1X1(128,4),X2(128,4),IT1(4),IT2(4)
      REAL K,K1,K2,K3
      IF(ICLASS.EQ.2)GO TO 30
      DO 10 I=1,4
      IT1(I)=IFIX(TIMEX(I))+1
      J=IT1(I)
      X1(J,I)=X(I)
      IF(TIMEX(I).LT.3.)GO TO 15
      II=IFIX(TIMEX(I)/2.)+1
      B=FLOAT(II)
      IB=IFIX(B)
      ID=J-1
      D=FLOAT(ID)
      K=(D+B+1.)/(D*B)
      GAMMA=(E1(I)-K)/(E1(I)+1.)
      UX(I)=UX(I)+GAMMA*(X(I)-UX(I))
      S=(X(I)*(D*(D+1.)-B*(B+1.))-X1(J-IB,I)*(D*(D+1.))+X1(1,I)
     1*(B*(B+1.)))/((D-B)*D*B)
      UX(I)=UX(I)+S
      K2=-(D+1.)/(B*(D-B))
      K3=(B+1.)/(D*(D-B))
      K1=(-(K2+K3)+1.)**2
      E1(I)=(E1(I)/(E1(I)+1.))*K1+K2**2+K3**2
   15 TIMEX(I)=TIMEX(I)+1.
   10 CONTINUE
      GO TO 60
   30 DO 40 I=1,4
      IT2(I)=IFIX(TIMEY(I))+1
      J=IT2(I)
```

```
      X2(J,I)=X(I)
      IF(TIMEY(I).LT.3.)GO TO 45
      II=IFIX(TIMEY(I)/2.)+1
      B=FLOAT(II)
      IB=IFIX(B)
      ID=J-1
      D=FLOAT(ID)
      K=(D+B+1.)/(D*B)
      GAMMA=(E2(I)-K)/(E2(I)+1.)
      UY(I)=UY(I)+GAMMA*(X(I)-UY(I))
      S=(X(I)*(D*(D+1.)-B*(B+1.))-X2(J-IB,I)*(D*(D+1.))+X2(1,I)
     1*(B*(B+1.)))/((D-B)*D*B)
      UY(I)=UY(I)+S
      K2=-(D+1.)/(B*(D-B))
      K3=(B+1.)/(D*(D-B))
      K1=(-(K2+K3)+1.)**2
      E2(I)=(E2(I)/(E2(I)+1.))*K1+K2**2+K3**2
   45 TIMEY(I)=TIMEY(I)+1.
   40 CONTINUE
   60 RETURN
      END
```

```
      SUBROUTINE PROCOV(CA,CB,CAINV,CBINV,DA,DB,X,ICLASS,IC,IFLAG)
C THIS SUBROUTINE MAKES A NEW ESTIMATE OF THE COVARIANCE OF A
C CLASS AFTER IC POINTS HAVE BEEN CLASSIFIED INTO THAT PARTICULAR
C CLASS.
      DIMENSION CA(16),CB(16),CAINV(16),CBINV(16),
     1PHIA(16),PHIB(16),X(4)
      IF(IFLAG.GT.0)GO TO 10
      ICNTA=1
      ICNTB=1
      DO 5 I=1,16
      CAINV(I)=CA(I)
      CBINV(I)=CB(I)
    5 CONTINUE
      CALL MINV(CAINV,4,DA)
      CALL MINV(CBINV,4,DB)
      GO TO 50
   10 IF(ICLASS.EQ.2)GO TO 30
      CALL COVAR(PHIA,X,ICNTA,4)
      ICNTA=ICNTA+1
      IF(ICNTA.LE.IC)GO TO 50
      DO 20 I=1,16
      CA(I)=PHIA(I)
      CAINV(I)=PHIA(I)
   20 CONTINUE
      CALL MINV(CAINV,4,DA)
      ICNTA=1
      GO TO 50
   30 CALL COVAR(PHIB,X,ICNTB,4)
      ICNTB=ICNTB+1
      IF(ICNTB.LE.IC)GO TO 50
      DO 40 I=1,16
      CB(I)=PHIB(I)
      CBINV(I)=PHIB(I)
   40 CONTINUE
      CALL MINV(CBINV,4,DB)
```

```
      ICNTB=1
   50 RETURN
      END
```

```
      SUBROUTINE COVAR(COV,X,INDEX,NDIM)
C THIS SUBROUTINE IMPLEMENTS THE RECURSIVE FORM OF COVARIANCE
C ESTIMATION.
      DIMENSION COV(4,4)X(4),XM(4)
      X1=FLOAT(INDEX)
      XO=X1-1.
      IF(INDEX.NE.1)GO TO 5
      DO 4 I=1,NDIM
      XM(I)=X(I)
      DO 3 J=1,NDIM
      COV(I,J)=0.
    3 CONTINUE
    4 CONTINUE
      GO TO 20
    5 DO 10 I=1,NDIM
      DO 11 J=1,NDIM
      COV(I,J)=(1./X1)*(XO*COV(I,J)+XO*XM(I)*XM(J)+X(I)*X(J))
     1-(1./(X1*X1))*(XO*XM(I)+X(I))*(XO*XM(J)+X(J))
   11 CONTINUE
   10 CONTINUE
      DO 15 I=1,NDIM
   15 XM(I)=(1./X1)*(XO*XM(I)+X(I))
   20 RETURN
      END
```

```
      SUBROUTINE MINV(A,N,D)
C THIS SUBROUTINE FINDS THE INVERSE OF A GENERAL MATRIX
C 'A', WHICH IS DESTROYED IN COMPUTATION.  THE INVERSE
C IS RETURNED IN 'A'.  THE DETERMINANT IS CALCULATED,
C 'D'.  'N' IS THE ORDER OF THE MATRIX, 'L' IS A WORK
C VECTOR OF LENGTH 'N', AND 'M' IS A WORK VECTOR OF
C LENGTH 'N'.
      DIMENSION A(16),L(4),M(4)
      D=1.0
      NK=-N
      DO 80 K=1,N
      NK=NK+N
      L(K)=K
      M(K)=K
      KK=NK+K
      BIGA=A(KK)
      DO 20 J=K,N
      IZ=N*(J-1)
      DO 20 I=K,N
      IJ=IZ+I
   10 IF (ABS(BIGA)-ABS(A(IJ))) 15,20,20
   15 BIGA=A(IJ)
      L(K)=I
      M(K)=J
   20 CONTINUE
      J=L(K)
      IF(J-K) 35,35,25
   25 KI=K-N
      DO 30 I=1,N
      KI=KI+N
      HOLD=-A(KI)
      JI=KI-K+J
      A(KI)=A(JI)
   30 A(JI)=HOLD
   35 I=M(K)
```

```
      IF(I-K) 45,45,38
   38 JP=N*(I-1)
      DO 40 J=1,N
      JK=NK+J
      JI=JP+J
      HOLD=-A(JK)
      A(JK)=A(JI)
   40 A(JI)=HOLD
   45 IF(BIGA) 48,46,48
   46 D=0.
      RETURN
   48 DO 55 I=1,N
      IF(I-K) 50,55,50
   50 IK=NK+I
      A(IK)=A(IK)/(-BIGA)
   55 CONTINUE
      DO 65 I=1,N
      IK=NK+I
      HOLD=A(IK)
      IJ=I-N
      DO 65 J=1,N
      IJ=IJ+N
      IF(I-K) 60,65,60
   60 IF(J-K) 62,65,62
   62 KJ=IJ-I+K
      A(IJ)=HOLD*A(KJ)+A(IJ)
   65 CONTINUE
      KJ=K-N
      DO 75 J=1,N
      KJ=KJ+N
      IF(J-K) 70,75,70
   70 A(KJ)=A(KJ)/BIGA
   75 CONTINUE
      D=D*BIGA
```

```
      A(KK)=1.0/BIGA
   80 CONTINUE
      K=N
  100 K=K-1
      IF(K) 150,150,105
  105 I=L(K)
      IF(I-K) 120,120,108
  108 JQ=N*(K-1)
      JR=N*(I-1)
      DO 110 J=1,N
      JK=JQ+J
      HOLD=A(JK)
      JI=JR+J
      A(JK)=-A(JI)
  110 A(JI)=HOLD
  120 J=M(K)
      IF(J-K) 100,100,125
  125 KI=K-N
      DO 130 I=1,N
      KI=KI+N
      HOLD=A(KI)
      JI=KI-K+J
      A(KI)=-A(JI)
  130 A(JI)=HOLD
      GO TO 100
  150 RETURN
      END
```

```
      SUBROUTINE MPRD(A,B,R,N,M,L)
C THIS SUBROUTINE FORMS THE PRODUCT OF TWO GENERAL MATRICES
C 'A' AND 'B', AND RETURNS THE PRODUCT IN 'R'.  'N' IS THE NUMBER
C OF ROWS IN 'A', 'M' IS THE NUMBER OF COLUMNS IN 'A' AND ROWS
C IN 'B', AND 'L' IS THE NUMBER OF COLUMNS IN 'B'.
      DIMENSION A(16),B(16),R(16)
      IR=0
      IK=-M
      DO 10 K=1,L
      IK=IK+M
      DO 10 J=1,N
      IR=IR+1
      JI=J-N
      IB=IK
      R(IR)=0
      DO 10 I=1,M
      JI=JI+N
      IB=IB+1
   10 R(IR)=R(IR)+A(JI)*B(IB)
      RETURN
      END
```

```
      SUBROUTINE OUTPUT (IY)
C THIS SUBROUTINE WRITES ONE 128 POINT TOW (BYTE DATA POINTS) OF A
C 128 X 128 POINT ARRAY FOR BOUNDARY DISPLAY VIA VIDEO DISPLAY SYSTEM.
      LOGICAL*1 ICDATA(128)
      COMMON /DUMB2/ICDATA
      WRITE(2'IY)ICDATA
      RETURN
      END
```